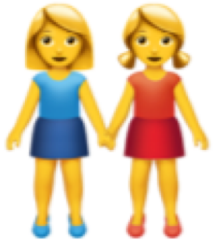
A Practical Intro to DevOps Practices and Tools

Chris Jackson
@chrijack

(In this Session)

Ideas



Compare

Sharpen Skills

Agenda

What is this DevOps Thing?

Why DevOps Matters

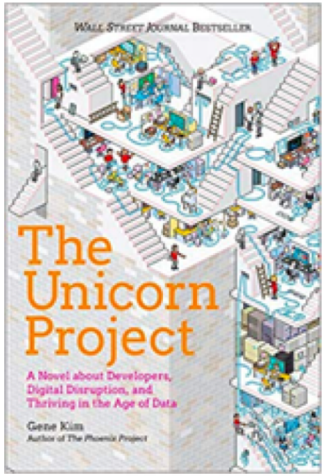
Going Faster with DevOps

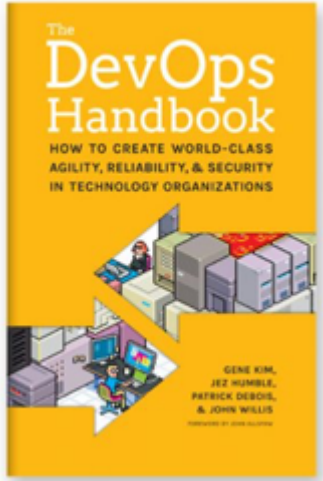
DevOps in Action

What is DevOps?

“[DevOps is] a set of cultural norms and technical practices that enable this fast flow of work from dev through test through operations while preserving world class reliability”

Gene Kim - author of “The Phoenix Project”, “DevOps Handbook”, and ”The Unicorn Project”





Pearson

What Does It Take to Embrace DevOps?

Culture	Agile/Lean	Business driven
Automation	Improve Speed and Frequency	Reduce effort
Measurement	Gather analytics	Everything is analyzed and visualized
Sharing	Shared Fate/Goals	

The ”3 Ways” of DevOps

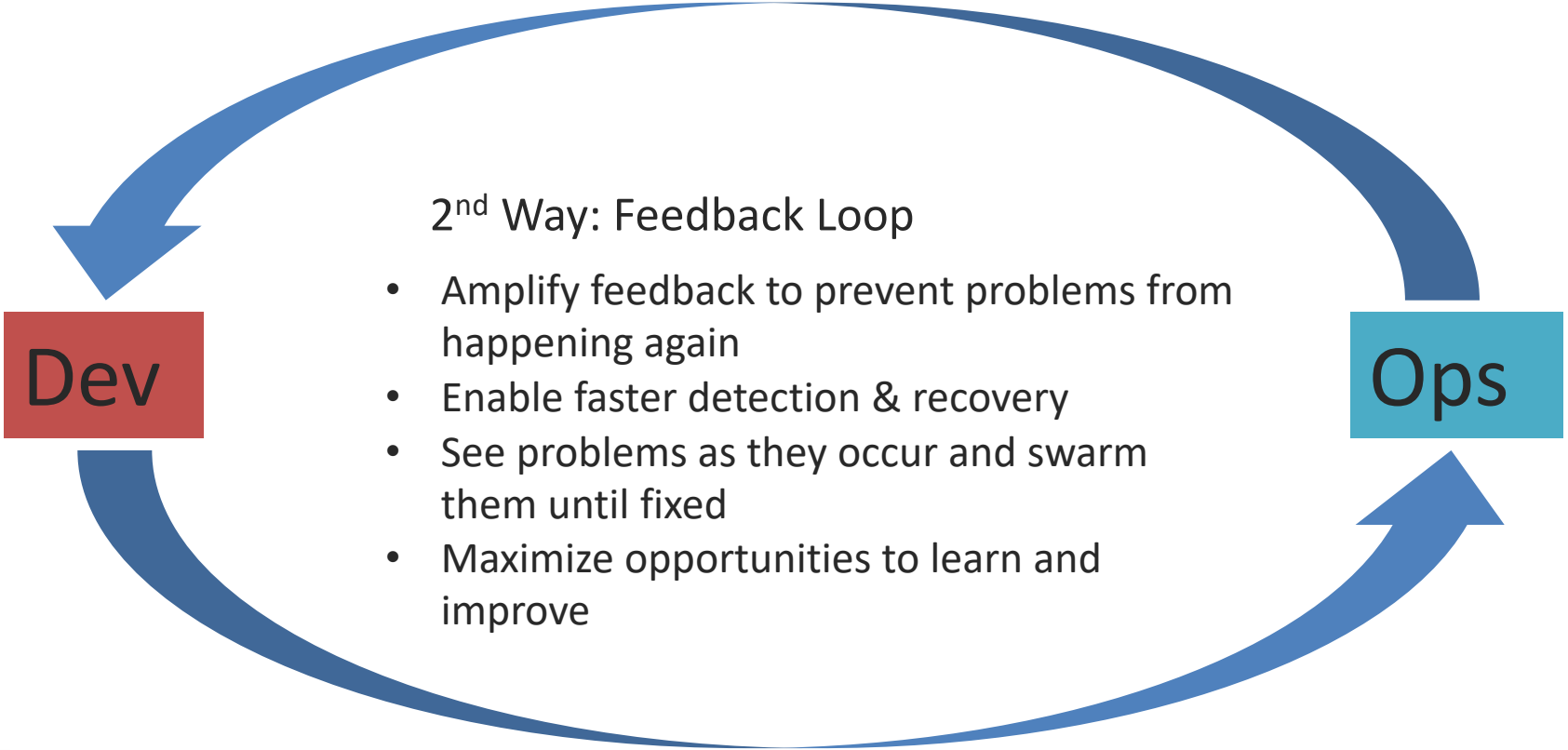
1st Way: Systems and Flow

Make work visible

- Reduce batch sizes
- Reduce intervals of work
- Build in quality by preventing defects from being passed downstream

Constantly optimize for business goals

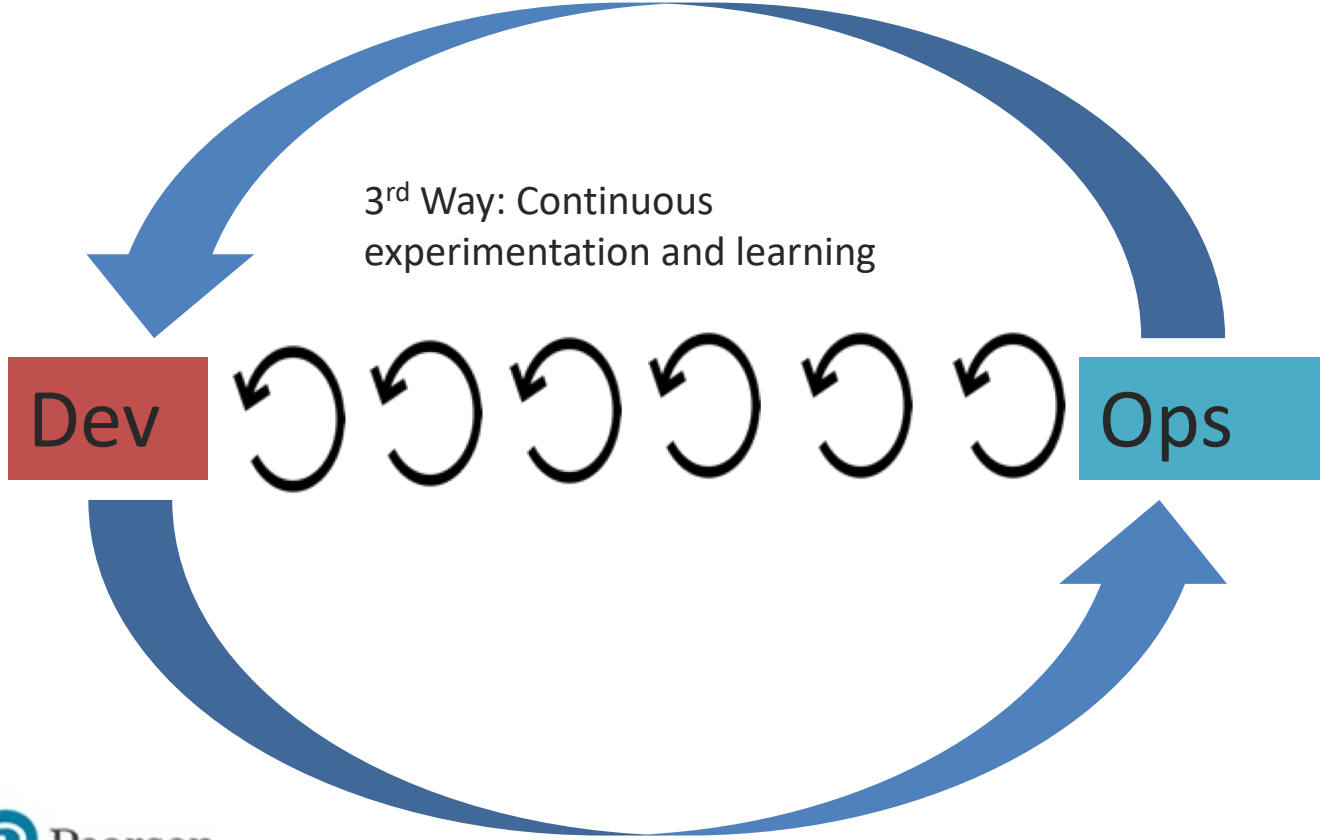
The ”3 Ways” of DevOps



2nd Way: Feedback Loop

- Amplify feedback to prevent problems from happening again
- Enable faster detection & recovery
- See problems as they occur and swarm them until fixed
- Maximize opportunities to learn and improve

The ”3 Ways” of Devops



- Conduct dynamic, disciplined, experimentation and risk-taking
- Define time to fix issues and make the system better
- When things go wrong don’t point fingers
- Create shared code repositories

Why DevOps Matters

Highest
Paid
Persons
Opinion

“Instead of trying to get better at predicting the future, we should improve our ability to adapt rapidly and effectively to new information”

Lean Enterprise: How High Performance Organizations Innovate at Scale

Improve Ability to Respond and Adapt

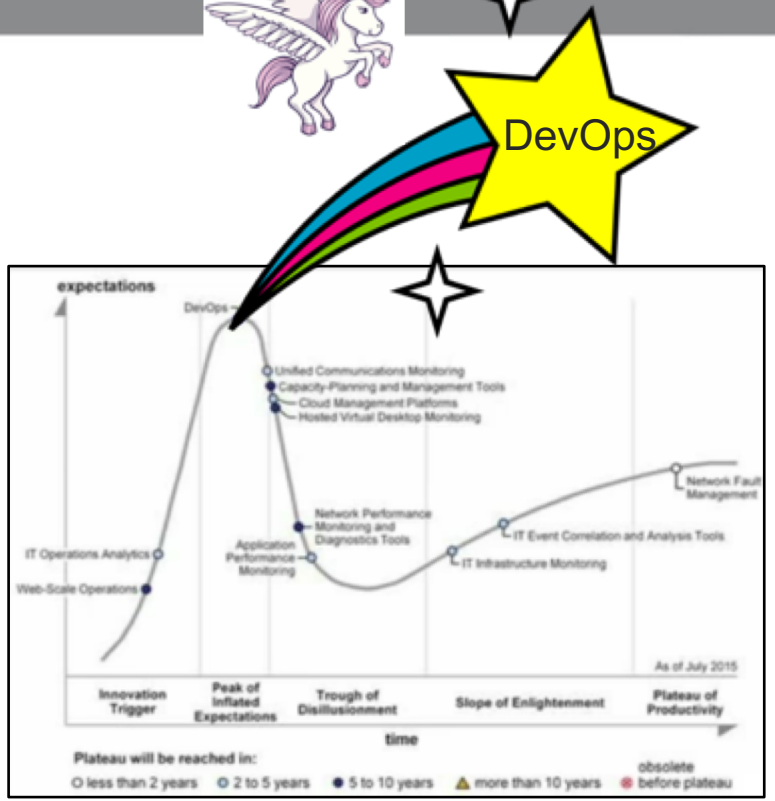
1. Your organization needs information to understand how to do better in the marketplace.

2. The more you experiment, the more information you can gather on what works

3. Solid Information leads to an improved strategy and benefits your customer as well as the business

DevOps Hype

- DevOps has reached maximum hype
- No longer for Silicon Valley startups
- Moving target - Is defined, but loose standards
- Casualties abound
- Many successes too!



Source: Gartner

DevOps Darlings

- 1000s of releases a day
- Fully automated build tools to test and make packages • Fully automated machine image bakery • Fully automated image deployment
- Developers deploy when they want • manage their own capacity and auto scaling • and fix anything that breaks

- Transformed a 186 year old bank into a technology company
- 3000 Deploys to production a month through PLATO platform
- React to customer feedback in REALTIME

- 90+ APIs facilitating in store experience, supply chain, and back office
- 100s of deployments a day
- Monthly API volume over 1.5 billion hits
- Less than 10 incidents a month
- Target DOJO

“If you are doing quarterly releases and your competitor is doing daily releases you will fall so far behind”

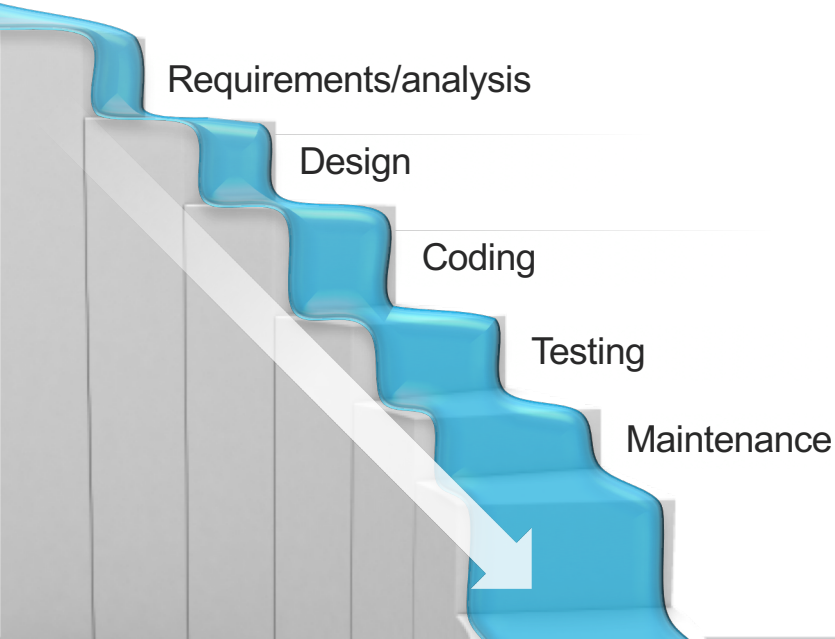
Adrian Cockcroft – former Netflix Architect

Waterfall Development Process

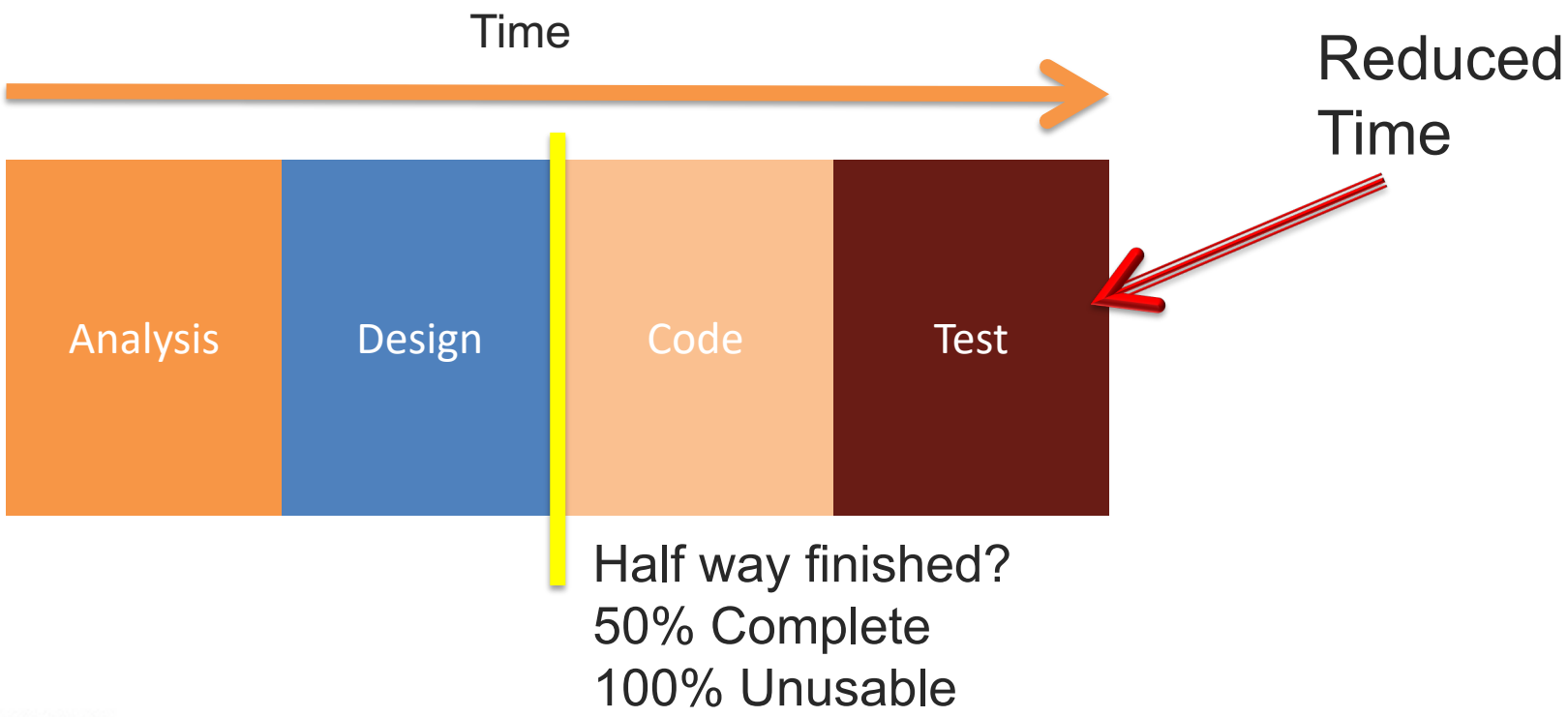
Around since the 50s

Sequential Design Approach

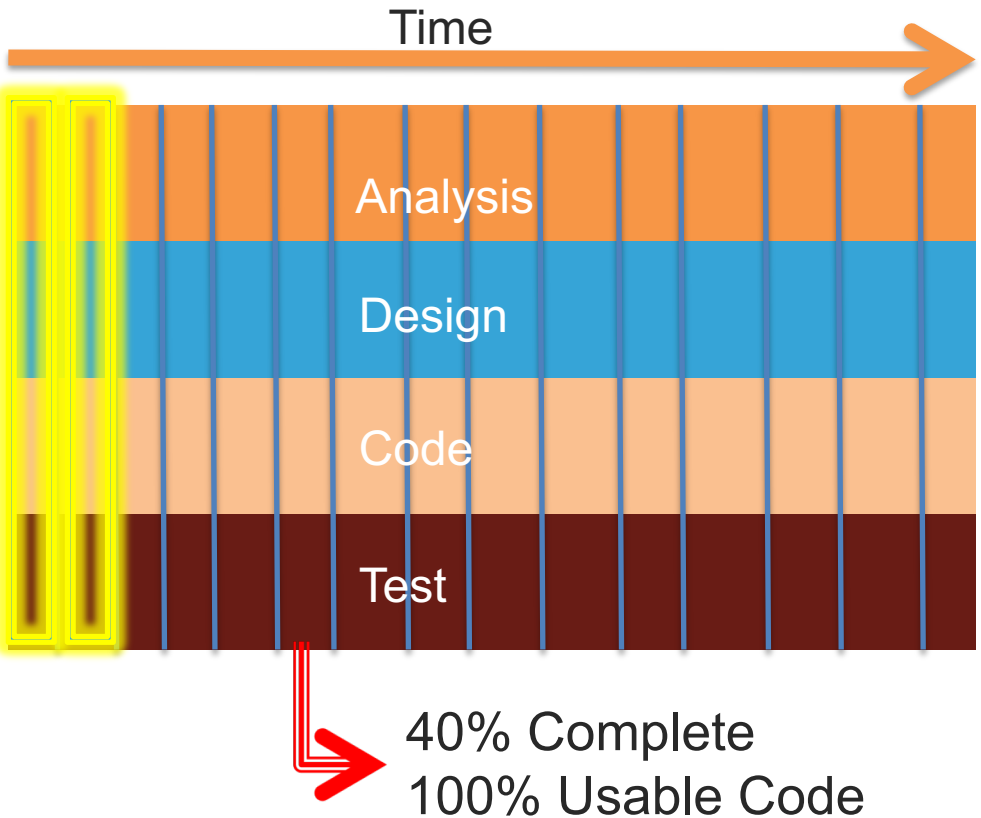
Requirements and scope are fixed



Before Agile Methodologies



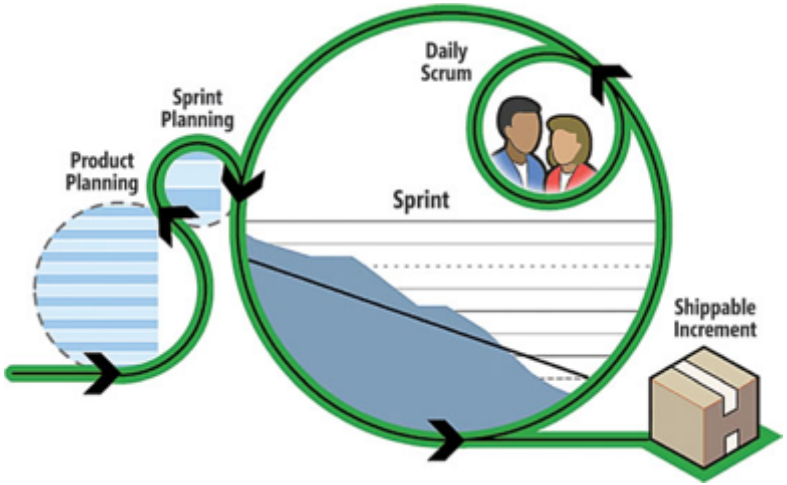
Benefits of Agile Development



Lean and Agile

Lean – Management philosophy to eliminate waste across all aspects of business

- Derived from Toyota Production System



Agile – Implementation of Lean for software development

- Short sprints

Operations World

- Care About
  - Everything is stable
  - Standards
  - Templates
  - Not getting bothered at 2:00 am
- Success
  - Software is stable
  - Backup and restore works
  - Systems are operating within defined thresholds

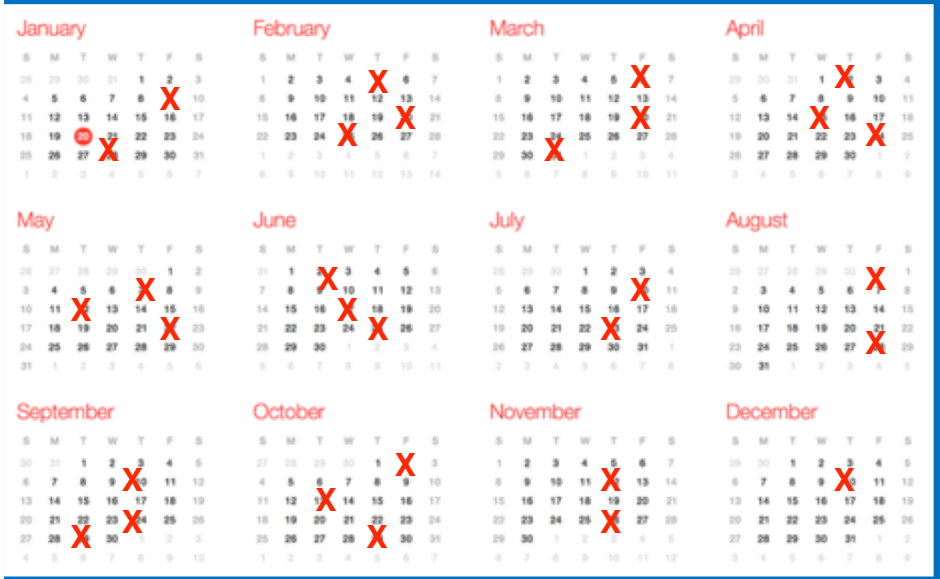
How Developers See Ops

Developers World

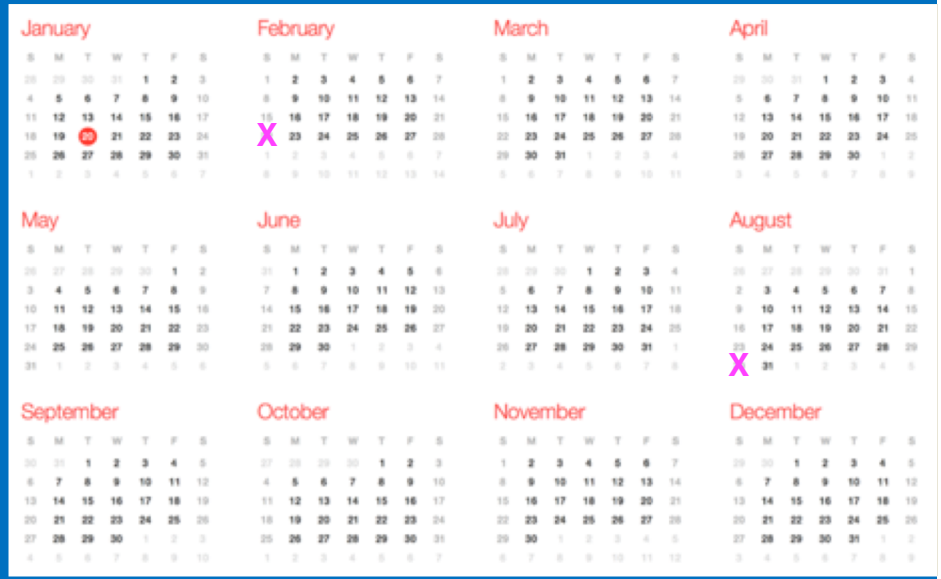
- Care About
  - Writing Software
  - Working Code
  - APIs
  - Libraries
  - Sprints
- Success
  - Software works – Laptop and Test
  - Finished Sprint

How Ops See Developers

Completed Sprints



Change Windows



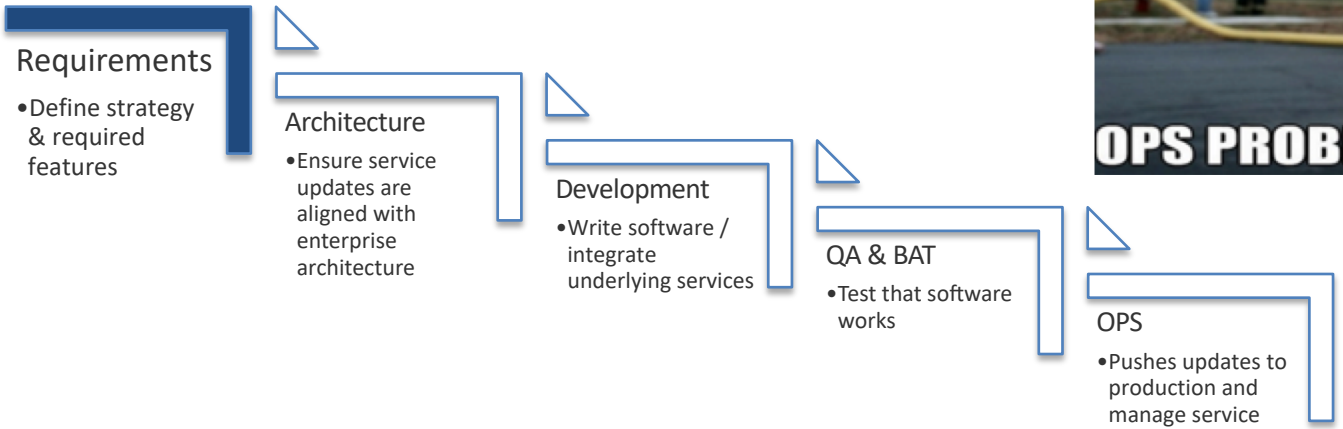
Change

Stability

Pearson

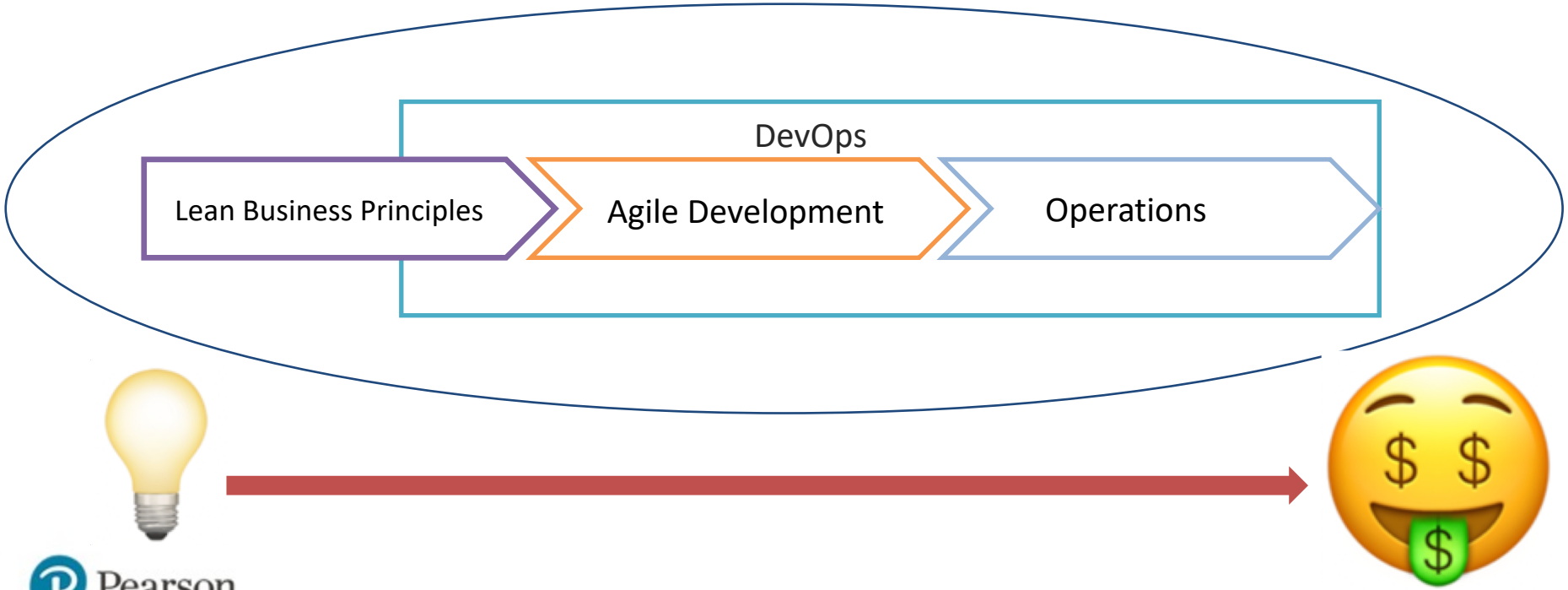
Traditional IT service delivery

- Slow, manual, and error prone



The more complex a project becomes, the longer the schedule, and the higher the probability of scope and schedule surprises.

Lean, Agile, and DevOps Combined



Click to edit Master title style

Break #1. 15 minutes

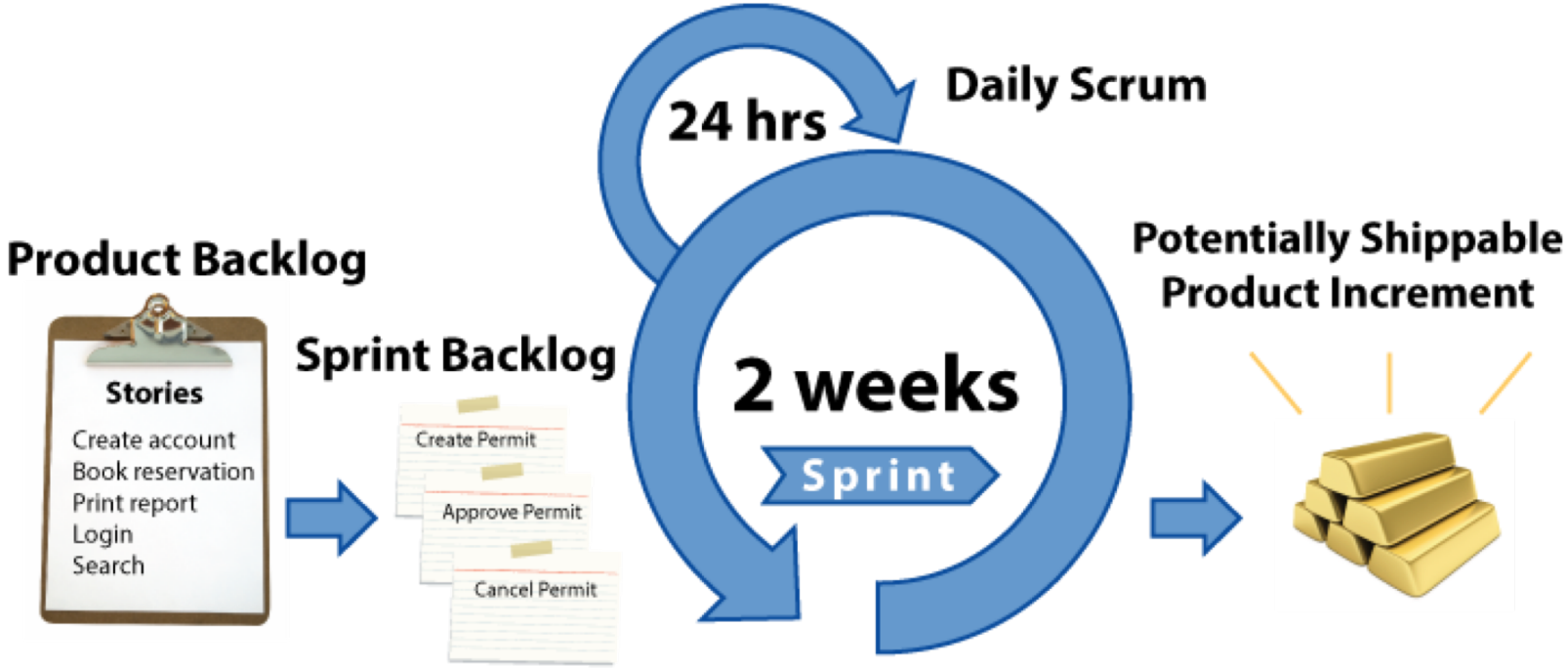
Going Faster with DevOps

Extreme Programing (XP)

- Focuses on Software Quality and Responding quickly to changing customer requirements
- Frequent releases and short dev cycles
- Improve productivity and interaction with customer

- Paired Programing
- Test Driven Development
- Continuous Integration
- Continuous Delivery

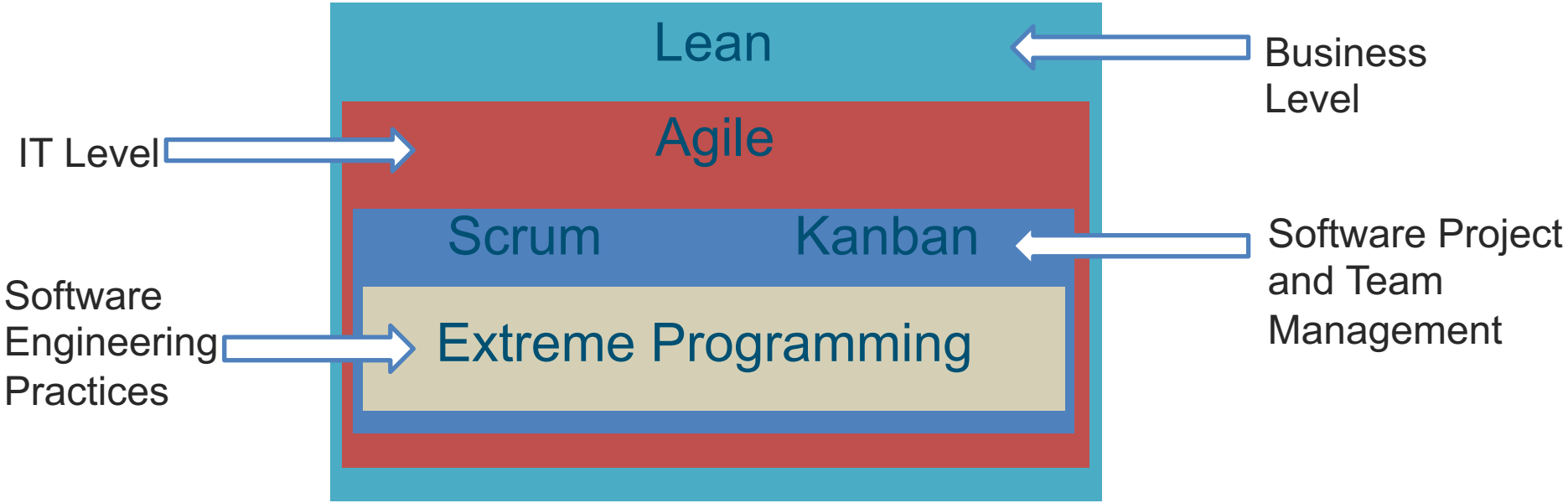
SCRUM Agile Project Management



Kanban

- Japanese for “Signal Board”
- 5 Core Principles
  - Visualize the workflow
  - Limit Work In Process
  - Manage Flow
  - Make Process Policies Explicit
  - Improve Collaboratively

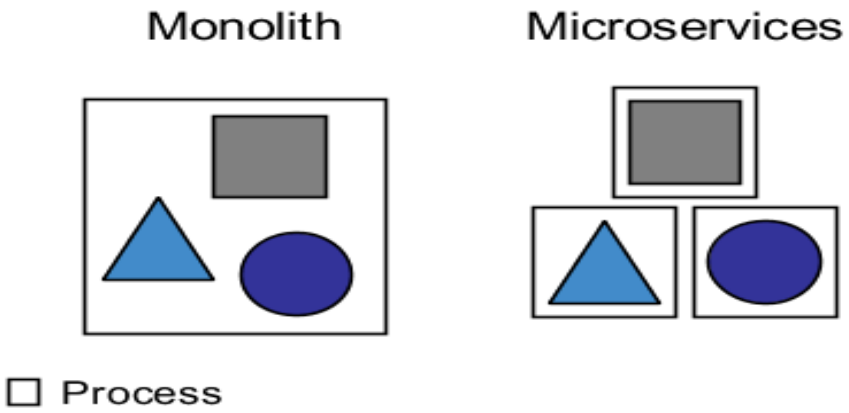
Pulling it all together



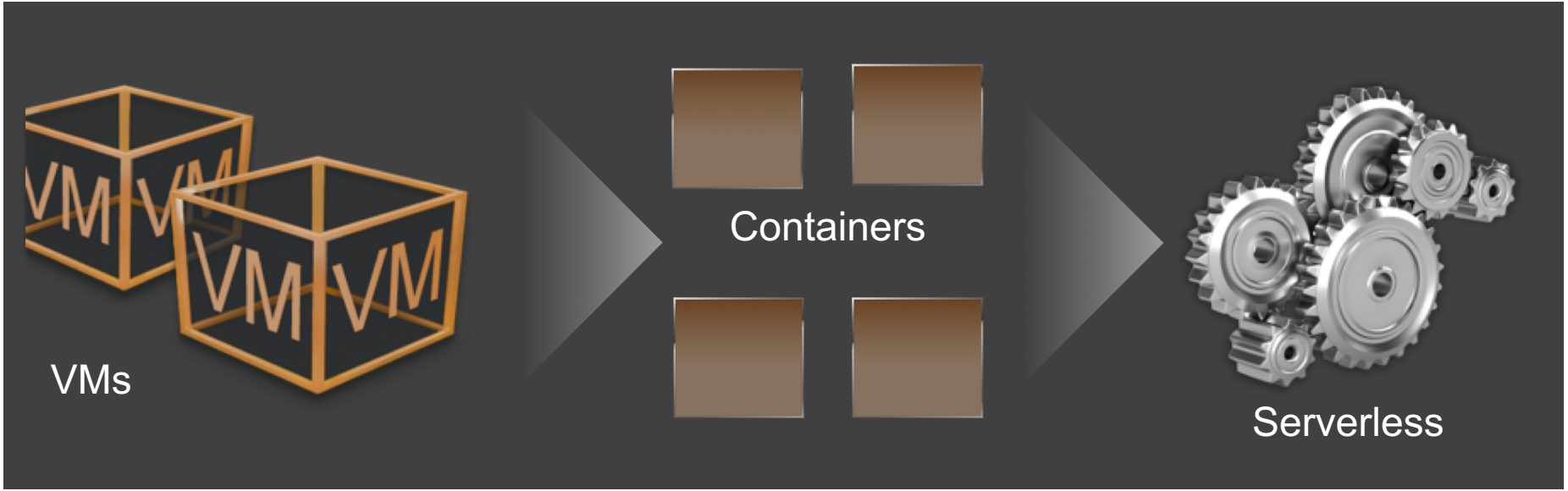
Make Operations Faster

What is a Microservice?

- A small and focused piece of software
- Stateless loosely coupled
- Language and technology independent
- Highly scalable and fault tolerant



Application Evolution is Driving Technology Adoption



Containers vs Virtual Machines

How are they different?

- Containers share a single OS kernel
- Containers tend to lock you into a single OS

When to use containers?

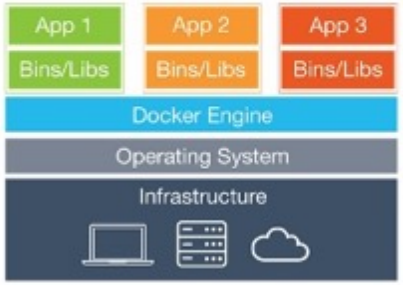
- When you want to run multiple instances of an app

When to use VMs?

- When you want to run multiple apps

…result is significantly faster deployment, much less overhead, easier migration, faster restart

Virtual Machines



Containers

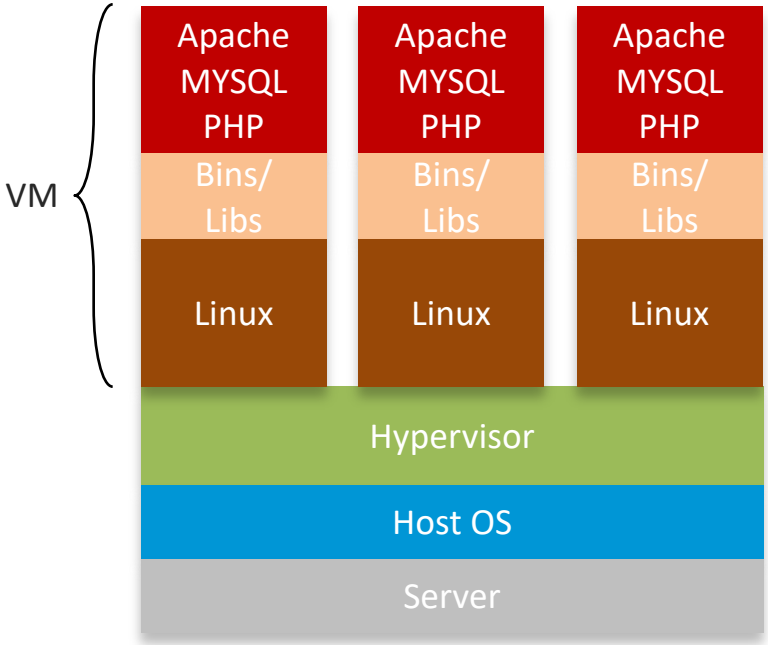
The Impact of Containers on DevOps

- Better aligning to the Dev and Ops of DevOps
  - Puppet, Chef, Ansible are popular with Ops but not Devs
  - Containers are equally popular with both
  - Containers create a natural segmentation of effort
    - Devs inside the container, Ops outside the container

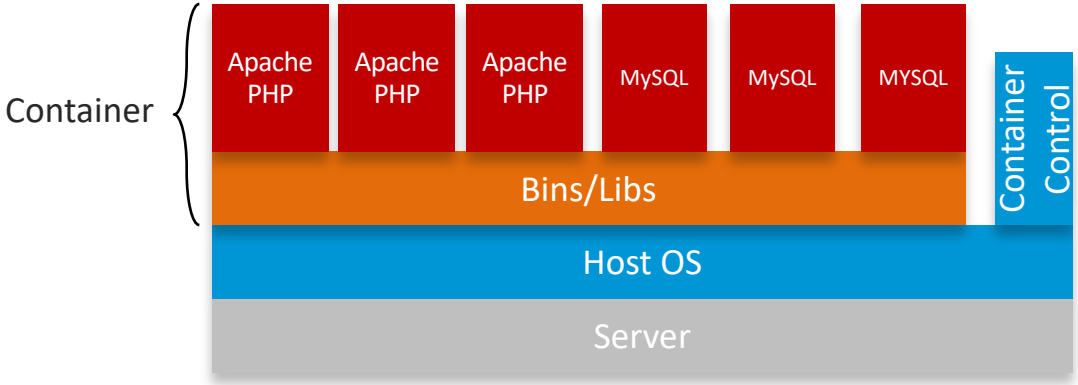
The Impact of Containers on DevOps

- Guaranteed consistency for CI/CD
  - Simplified light weight image files
    - Only what the app needs and nothing else
  - What is built on a laptop will work the same in production
- Open community built best of breed containers
  - Public container repository - Docker Hub
- SPEED
  - Deploy in seconds instead of minutes

“Containerizing” a Simple Web App



Containers are isolated but share OS and where appropriate bins/libraries



Understanding Docker

Docker allows you to package an application with all of its dependencies into a standardized unit for software development.

Docker containers wrap up a piece of software in a complete filesystem that contains everything it needs to run:

- code
- runtime
- system tools
- system libraries – *anything you can install on a server*

This guarantees that it will always run the same, regardless of the environment it is running in.

Docker Images, Registries, and Containers

Images *are read-only templates*. Images are used to create Docker containers.

There are public or private stores from which you upload or download images.

Containers *are similar to a directory.* A Docker container holds everything that is needed for an application to run.

Registries are stateless, scalable server side application that stores & lets you distribute Docker images.

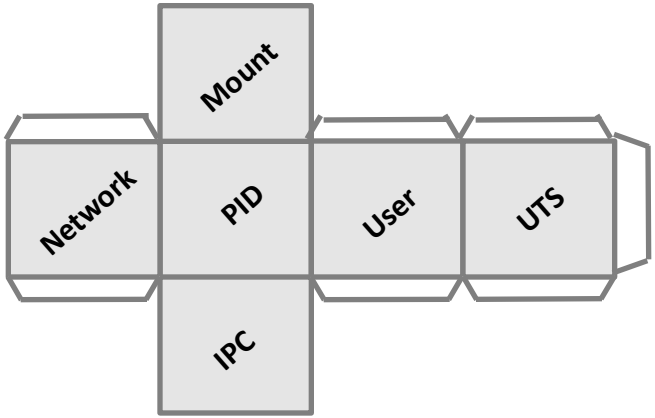
Think of **Docker Hub** as **Github** for **Docker Container Images**

Docker Hub
(SaaS, in the cloud)

Trusted Registry
(you manage, on-prem)

How Does a Container Work?

Namespaces provide processes with their own view of the system. Also limits what process can see (and therefore use).



- **pid**. isolate the process ID number space. Processes in different PID namespaces can have the same PID
- **net**. provide isolation of the system resources associated with networking
- **mount**. isolate the set of filesystem mount points seen by a group of processes.
- **uts**. allows each container to have its own hostname and NIS domain name
- **ipc**. isolate certain interprocess communication (IPC) resources
- **user**. isolate the user and group ID number spaces

Docker Architecture

- Client
  - CLI to Docker engine
  - Local or remote Docker engine
  - Uses RESTful API
- Docker Host
  - Runs Docker engine
  - Hosts containers
  - Stores images locally
- Docker Registry
  - Software distribution to hosts and engine
  - Docker Hub is public registry
  - Can be local, Public, Private

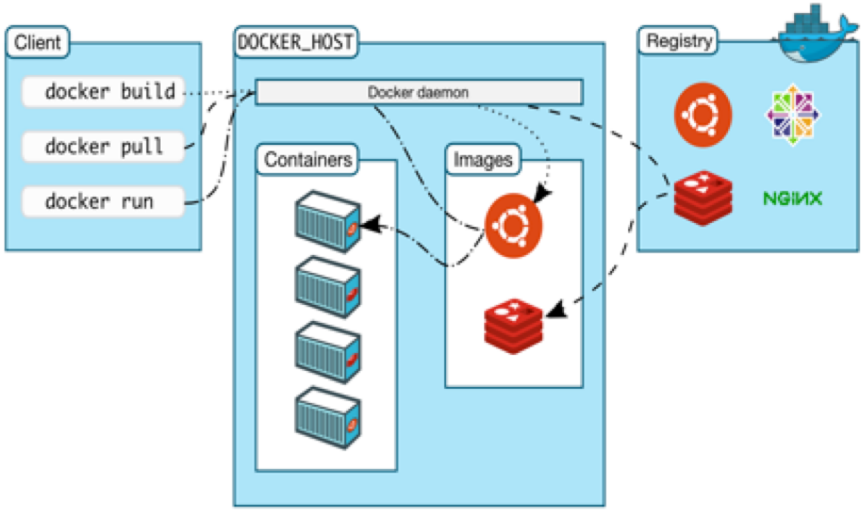
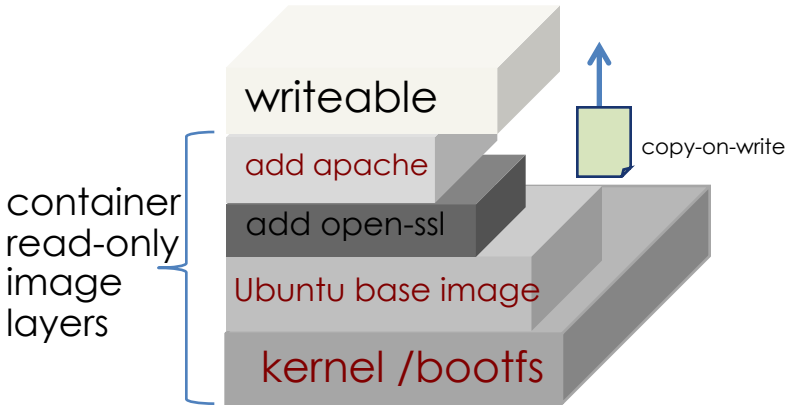
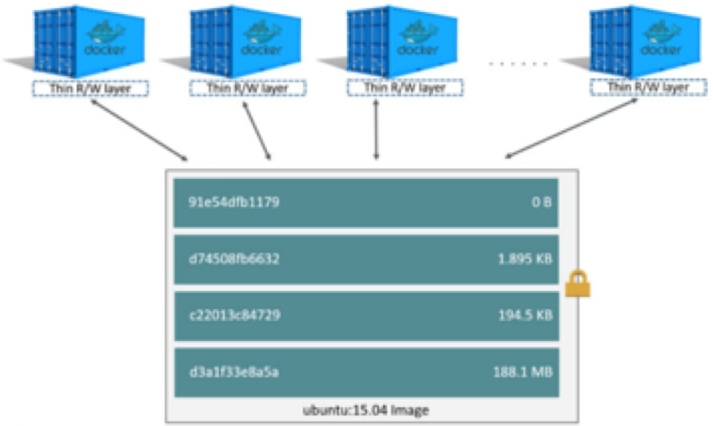


Image source:https://docs.docker.com/introduction/understanding-docker/

Docker Images

- **Images layered via union file system** – enables multiple layered file systems images to be seen as one image.





Docker Images – Union Mounts

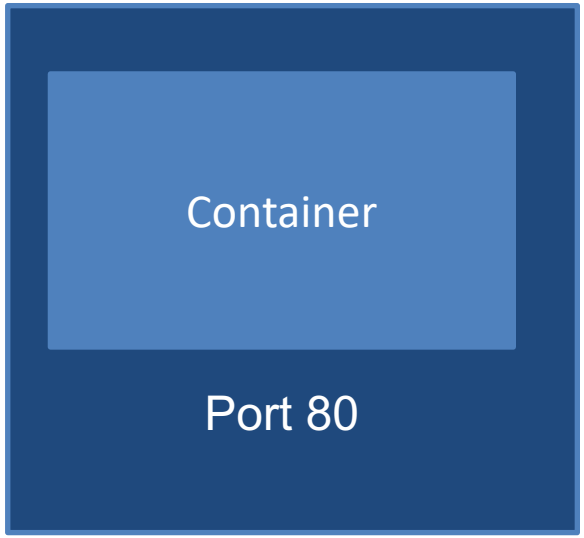
```
[root@Centos7 mnt]# docker images --no-trunc
REPOSITORY          TAG                 IMAGE ID                                                           CREATED                 VIRTUAL SIZE
docker.io/ubuntu    latest              8693db7e8a0084b8aacba184cfc4ff9891924ed2270c6dec6a9d99bdcff0d1aa   Less than a second ago  187.9 MB
[root@Centos7 mnt]#
[root@Centos7 mnt]# docker history 8693db7e8a0084b8aacba184cfc4ff9891924ed2270c6dec6a9d99bdcff0d1aa
IMAGE               CREATED                 CREATED BY                                      SIZE                COMMENT
8693db7e8a00        Less than a second ago  /bin/sh -c #(nop) CMD ["/bin/bash"]             0 B
a4c5be5b6e59        Less than a second ago  /bin/sh -c sed -i 's/^#\s*\(deb.*universe\)$/   1.895 kB
c4fae638e7ce        Less than a second ago  /bin/sh -c echo '#!/bin/sh' > /usr/sbin/polic   194.5 kB
f15ce52fc004        Less than a second ago  /bin/sh -c #(nop) ADD file:7ce20ce3daa6af21db   187.7 MB
[root@Centos7 mnt]# ls
8693db7e8a0084b8aacba184cfc4ff9891924ed2270c6dec6a9d99bdcff0d1aa  c4fae638e7ce5e7eb92b92e6d82e3180e28b85948d362dc52a1c44a720f9b52a
a4c5be5b6e596241b4530ade26294afa8d8a4a2b9163c30e36c87f879b0f5073  f15ce52fc004a5c3eab9128a78f7c0c2135d4f726bc54f1373120ab3ff291bcc
[root@Centos7 mnt]#
```

Demo: Docker

Click to edit Master title style

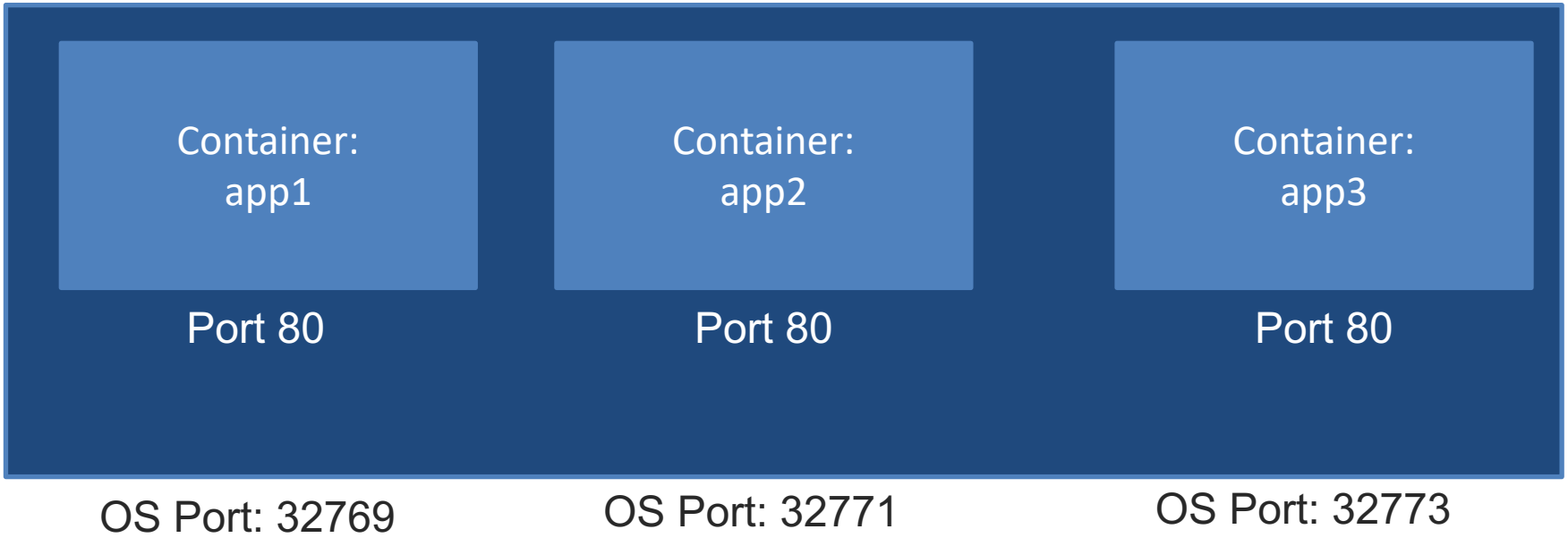
Break #2. 15 minutes

```
docker run –p 80:80 –d –name app app
```

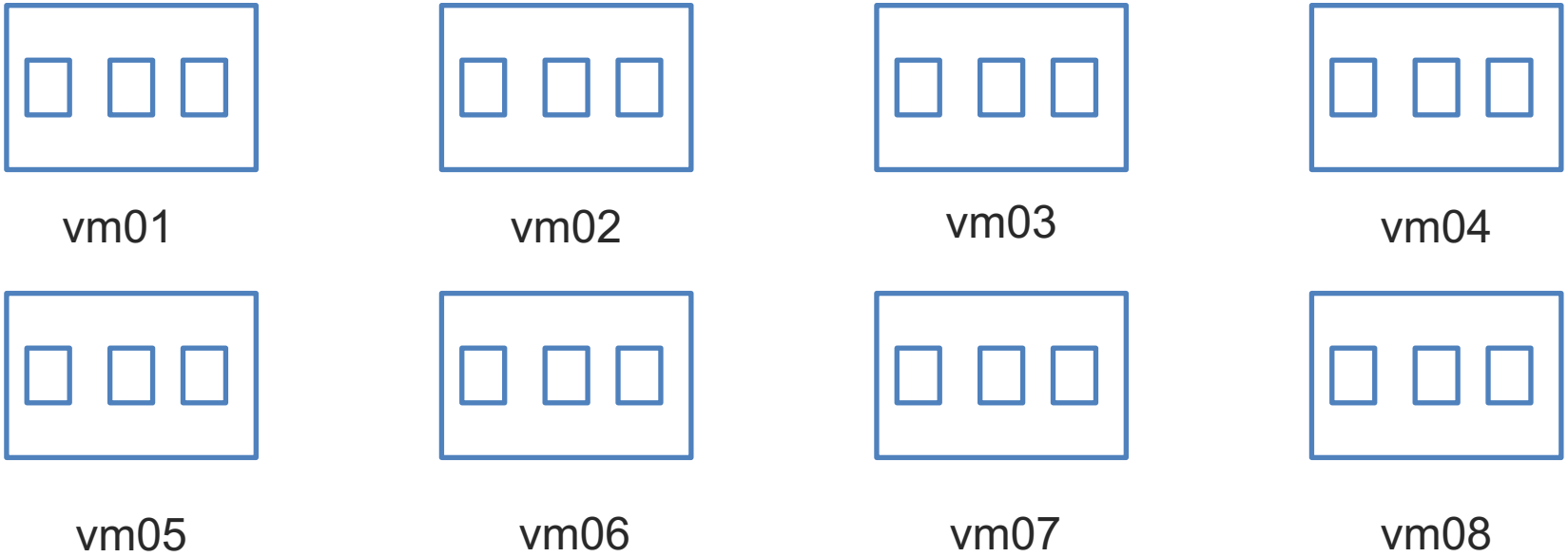


Operating System Port 80

```
for i in $(seq 3); do docker run –P –d –name app$i app; done
```

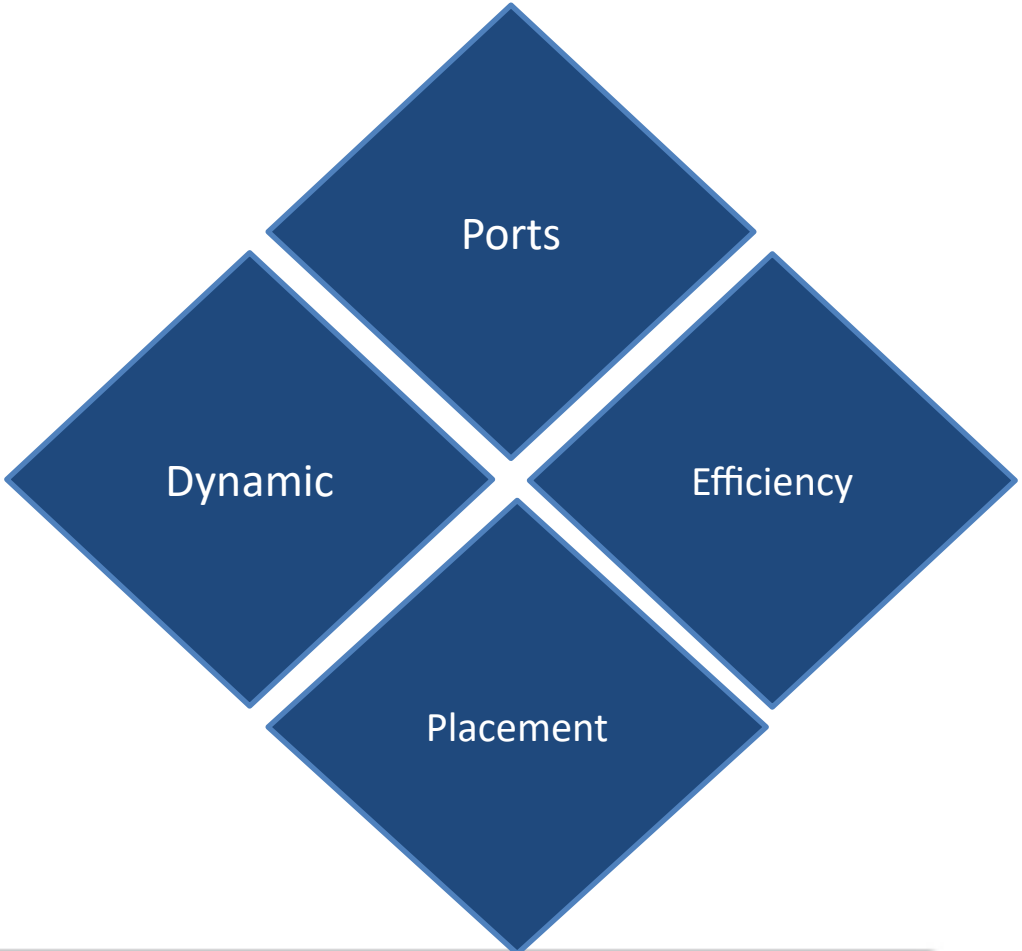


```
for j in $(seq -w 8); do ssh vm0$j for i in $(seq 3); \
do docker run -P -d -name app$j$i app; done; done
```



Pearson

4 Challenges of Containers

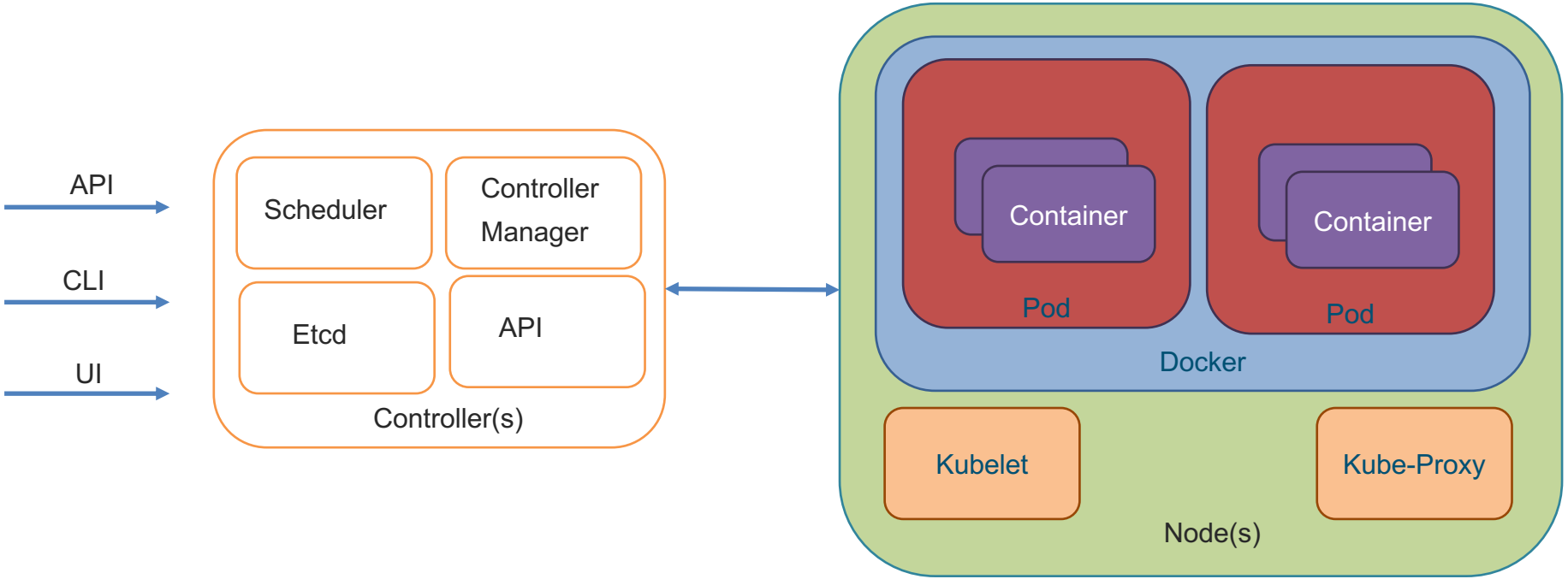


Solutions Emerge

What is Kubernetes (K8S) ?

- Open Source system started by Google in 2014 for automating deployment, operations, and scaling of containerized applications
- Written in Go (aka golang)
- Portable
  - Public, Private and Hybrid
- Extensible
  - Modular, Pluggable
- Self-Healing
  - Auto-scaling, Auto-replication
- Contributed to CNCF in July 2015

Kubernetes Deployment



Kubernetes in 360 S

Create a SERVICE

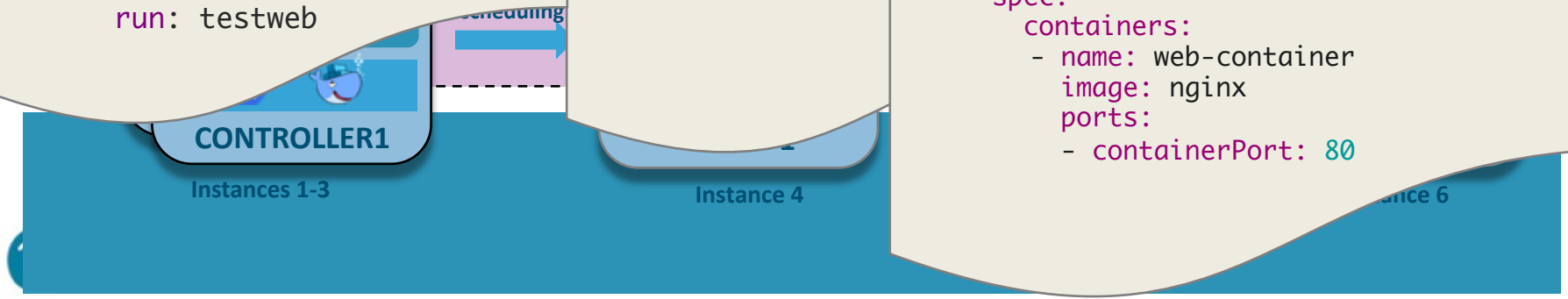
```
user@ubuntu:~/svc$ cat svc.yaml
apiVersion: v1
kind: Service
metadata:
 name: testweb
 labels:
   name: testweb
spec:
   type: ClusterIP
   ports:
   - port: 80
  selector:
     run: testweb
```

Create a Simple

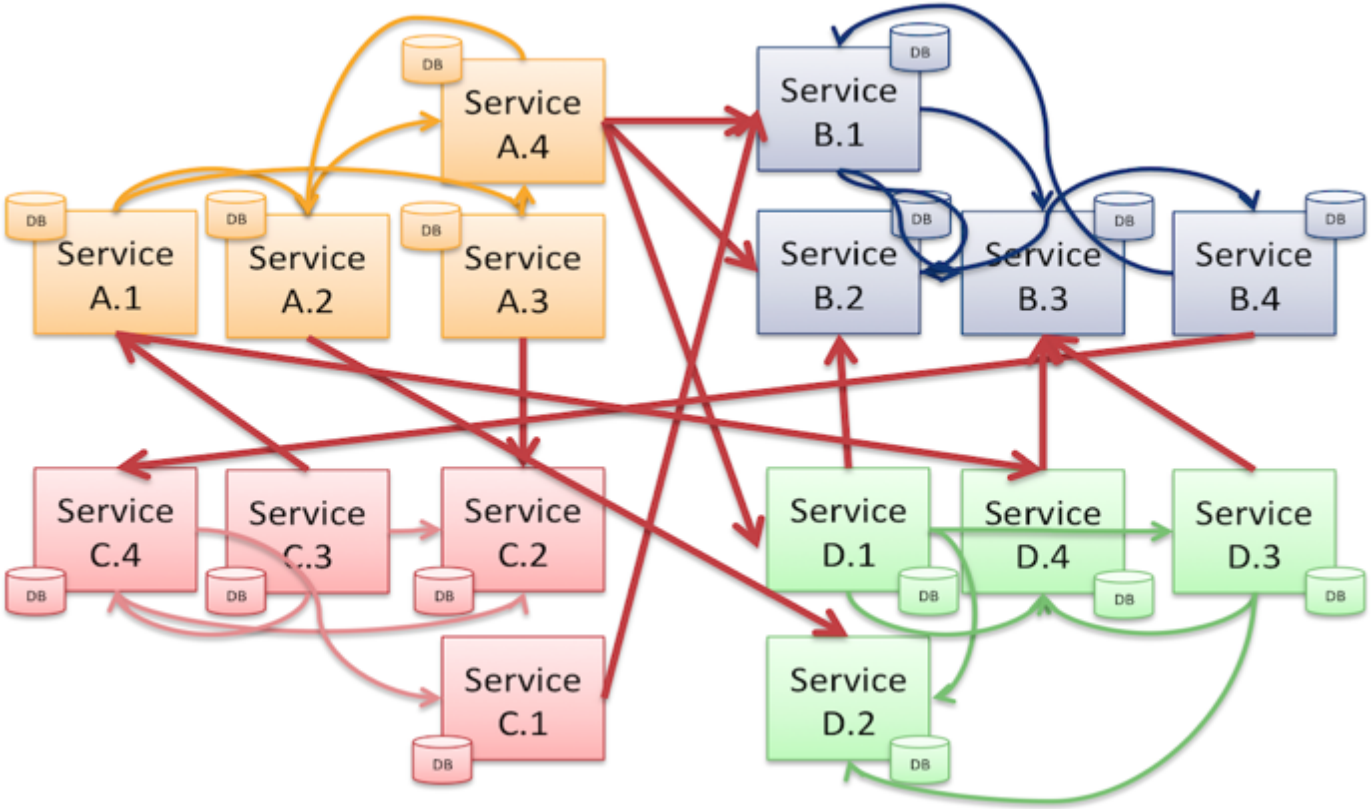
```
user@ubuntu:~/svc
apiVersion: v1
kind: Pod
metadata:
  name: bigwebstu
  labels:
    run: testweb
spec:
  containers:
  - name: web-cor
    image: nginx
    ports:
    - containerPo
```

Create PODs with a REPLICATION CONTROLLER

```
user@ubuntu:~/svc$ cat webrc.yaml
apiVersion: v1
kind: Deployment
metadata:
  name: bigwebstuff
  labels:
    name: bigwebstuff
spec:
  replicas: 2
  selector:
    run: testweb
  template:
    metadata:
      labels:
        run: testweb
   spec:
     containers:
     - name: web-container
       image: nginx
       ports:
       - containerPort: 80
```



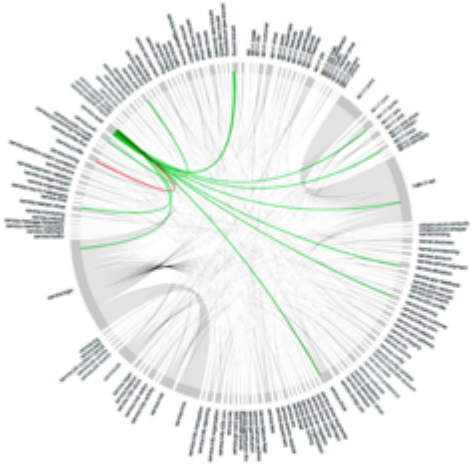
Microservice = LOTS of east west traffic



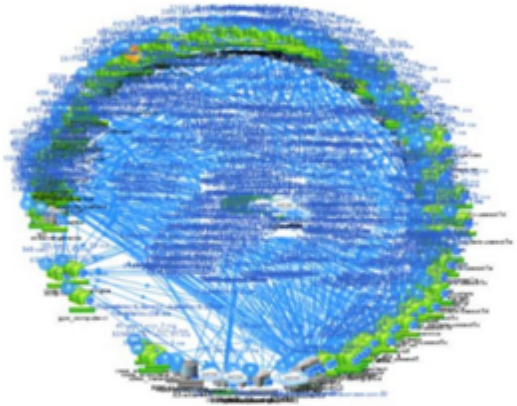
Pearson

Operating the Death Star

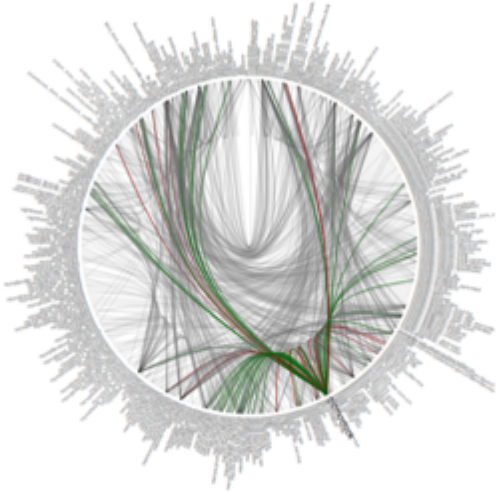
450+ microservices



500+ microservices



500+ microservices

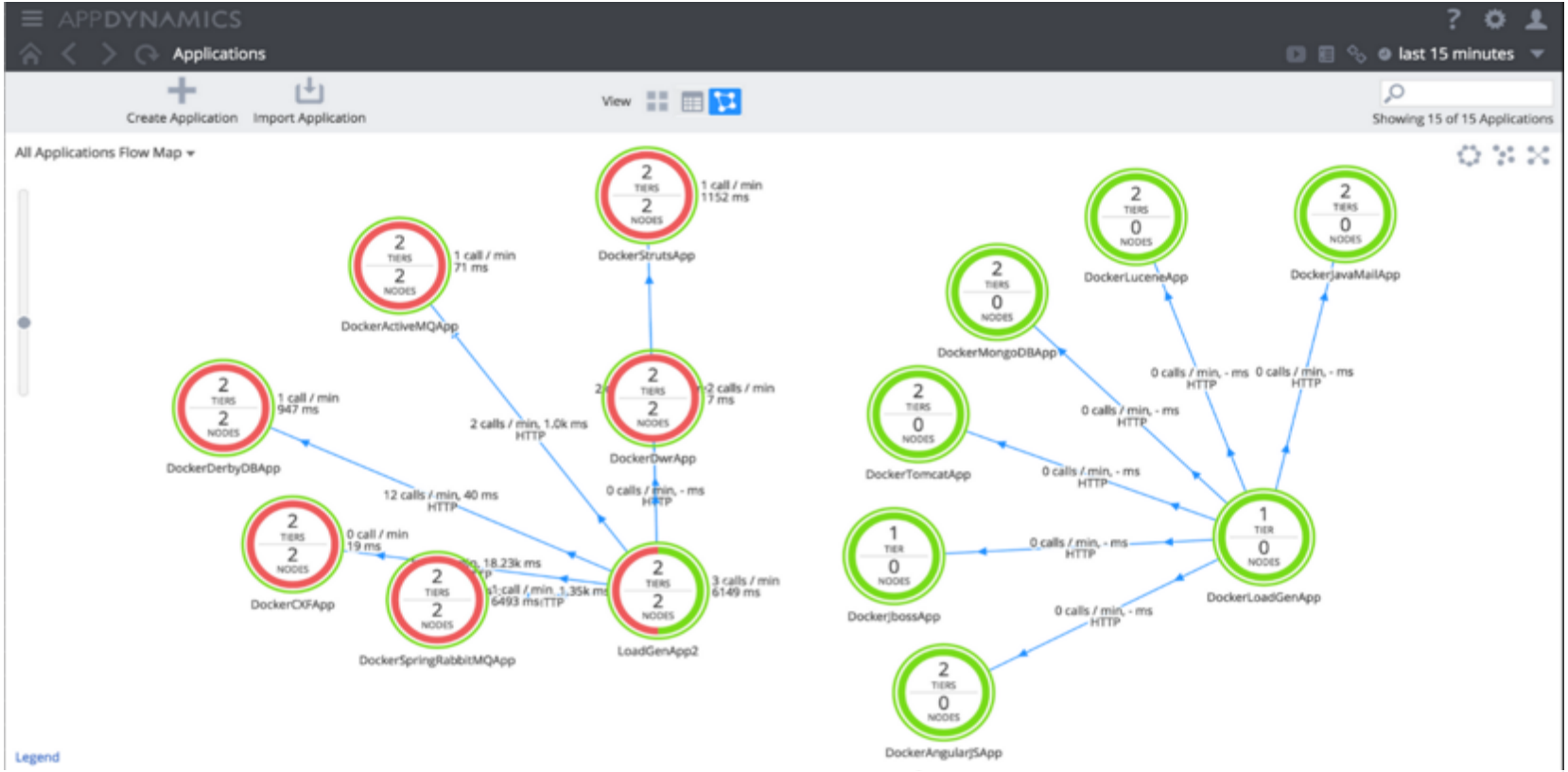


HAIL O

NETFLIX

Pearson

Operating Microservice Apps is Not



Pearson

Infrastructure as Code

(Configuration Management)

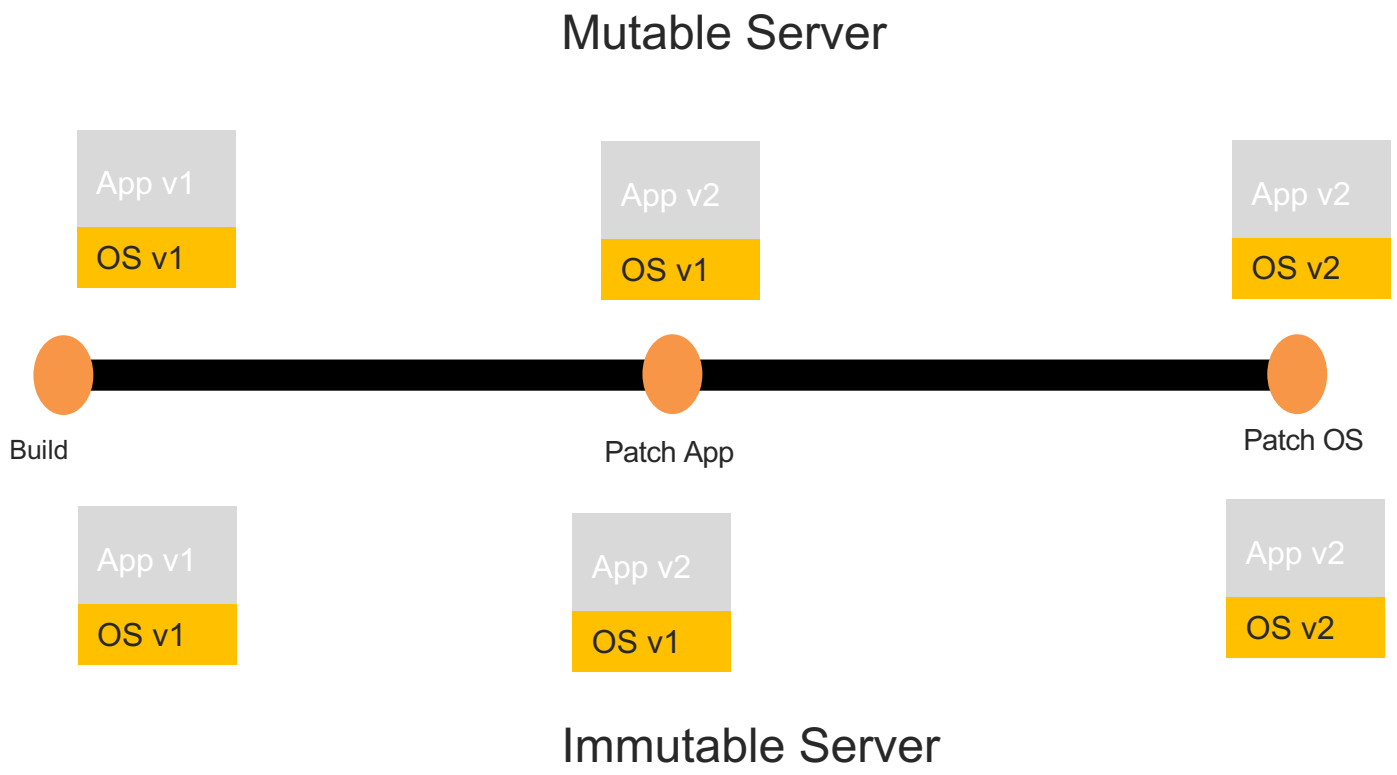
- Writing high level code that automates the provisioning and deployment of infrastructure components
- Not just script writing or infrastructure automation
- Uses software development practices
  - Versioning control
  - Design Patterns
  - Testing
- Infrastructure is DEFINED by the code
- Vagrant, Puppet, Chef, Ansible, Docker, ect

Immutable Infrastructure

IMMUTABLE : Unchanging over time or unable to be changed

- Once you instantiate something, it never changes
- Instead you replace it with a newly built instance
- Only works in a true cloud environment with API control over all aspects of Configuration and Monitoring

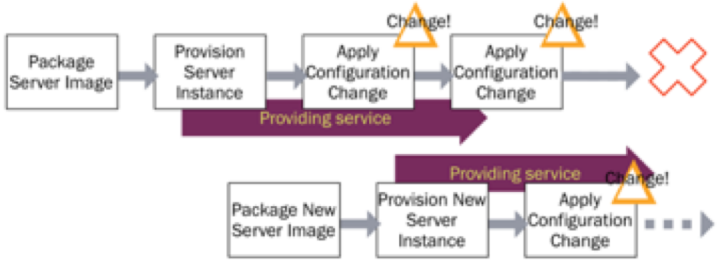
Immutable Infrastructure Stack

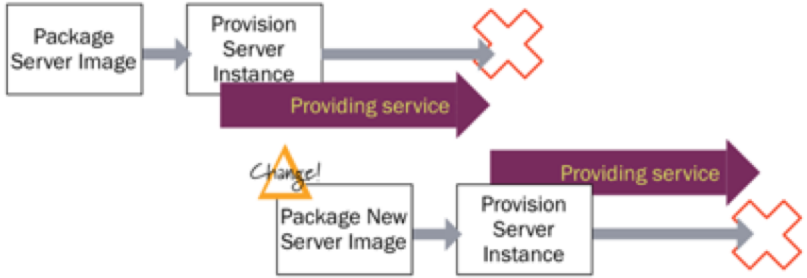


Traditional VM vs Container Delivery

- VM = 2-5 minutes
- Software Entropy
- Infrastructure as code upwards of 10 minutes to restore

- Container 500ms
- New instantiation each time with latest packages
- Immutable delivery model 2-3 secs





Cisco Container Platform

CLOUD NATIVE COMPUTING FOUNDATION

certified kubernetes

Turnkey Solution
For Production-Grade Optimized
Container Environments

Native Kubernetes (100% Upstream)
Direct updates and best practices from open source community

Hybrid Cloud Optimized
E.g: Google, Istio, external secure registry, …

Integrated
Networking | Management | Security | Analytics

Flexible Deployment Model
VM | Bare metal ←→ HX, ACI | Public cloud

Easy to acquire, deploy & manage | Open & consistent | Extensible platform | World-class advisory & support

Demo: Kubernetes

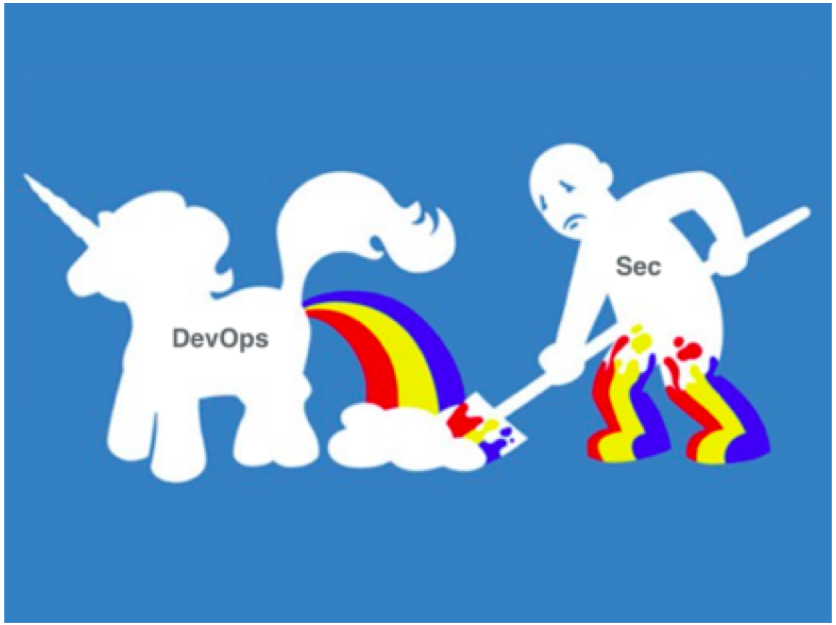
Click to edit Master title style

Break #3. 15 minutes

Go Fast Safely

What About Security?

- Lots of Dev and Ops, no mention of security
- Configuration management tools (Puppet, Chef, Ansible) can help standardize controls but not replace security expertise
- Must be inserted into the DevOps process (not a bolt on!)



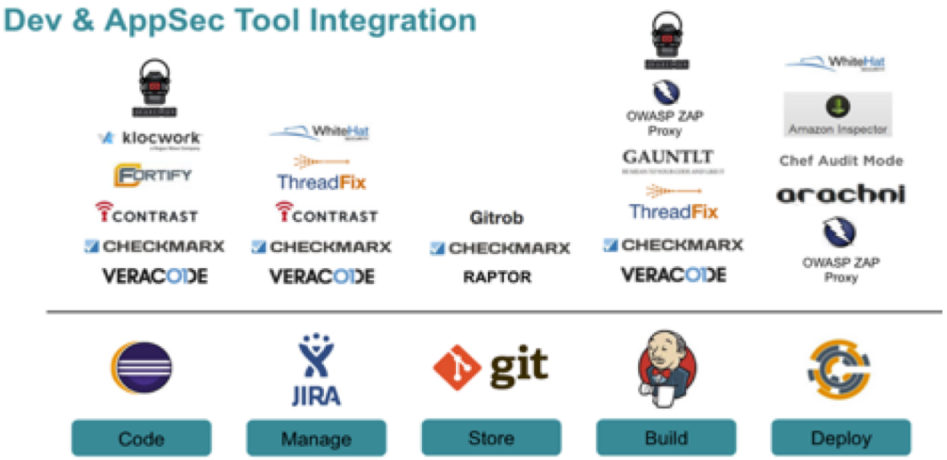
DevSecOps Efforts

- Rugged Software – Integrating Security into software development
- Gauntlt

GAUNTLT

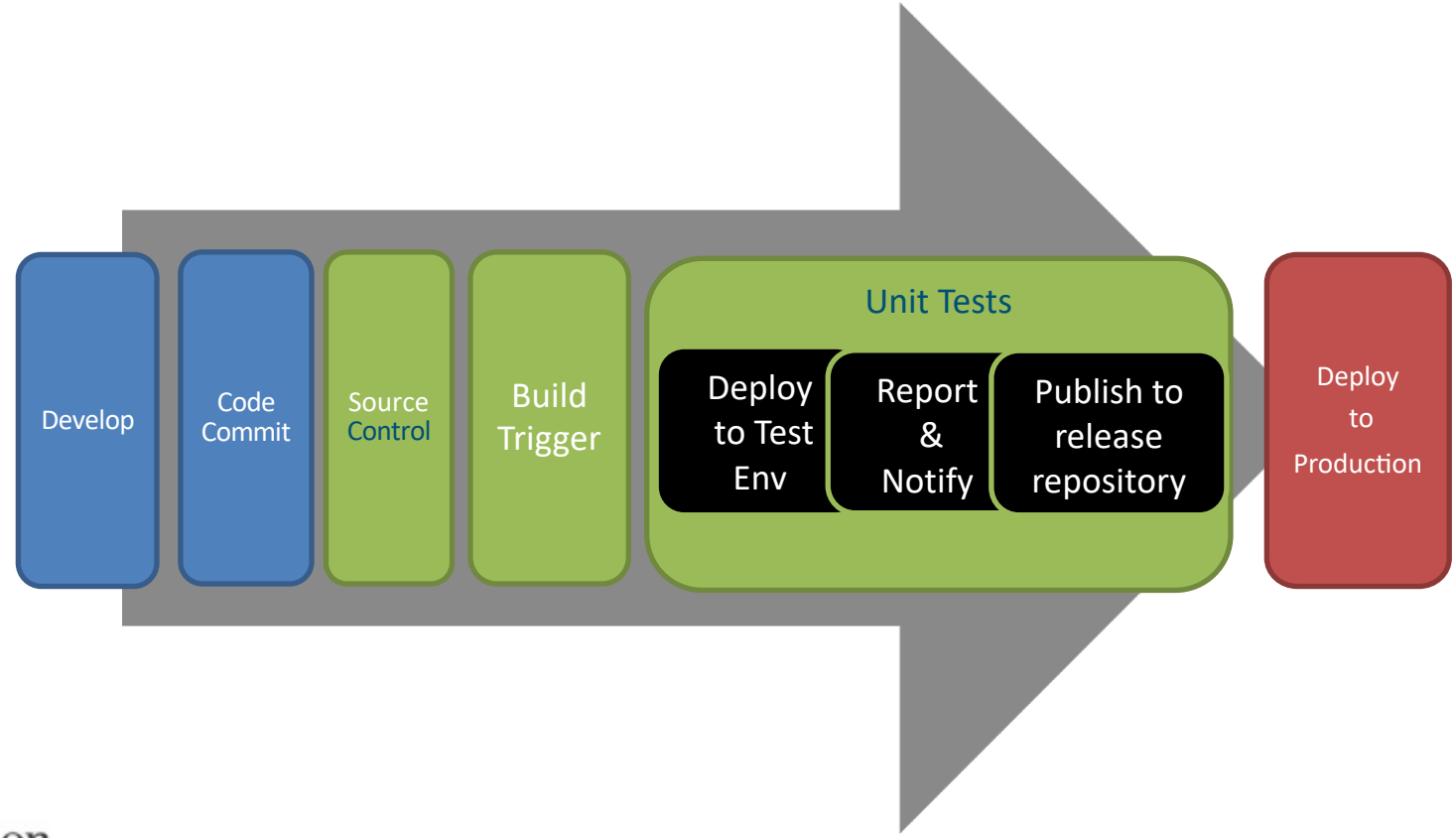
BE MEAN TO YOUR CODE AND LIKE IT

- OWASP AppSec Pipeline Project

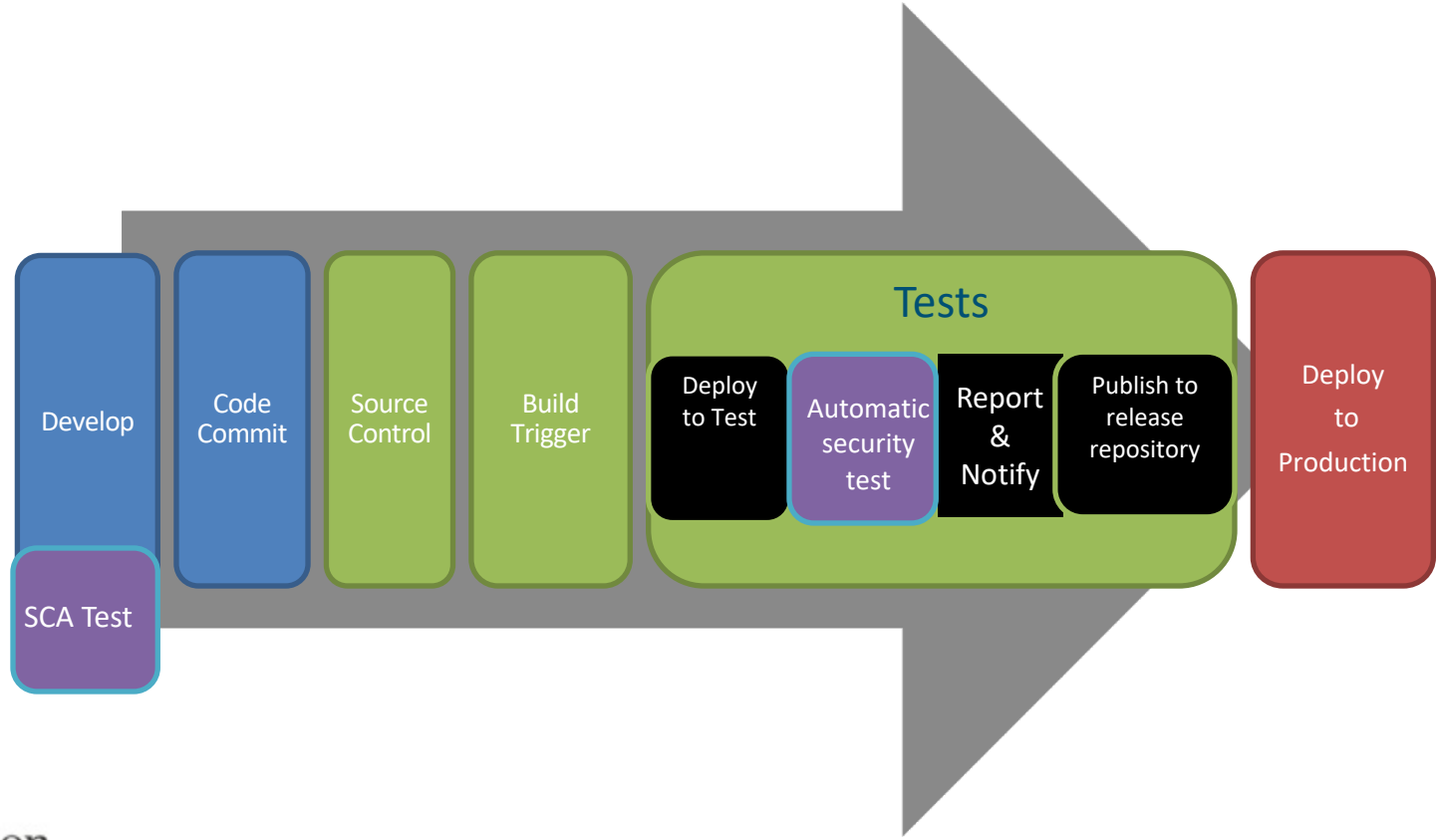


https://www.owasp.org/index.php/OWASP_AppSec_Pipeline

Continuous Deployment



Security Within Continuous Deployment



TLS all the things

- TLS/SSL can now be free
- Encrypt between Microservices

DevOps Pipeline Tools

DevOps Periodic Table

PERIODIC TABLE OF DEVOPS TOOLS (V3)

EMBED DOWNLOAD ADD

Os	Open Source
Fr	Free
Fm	Freemium
Pd	Paid
En	Enterprise

Source Control Mgmt.	Deployment	Analytics
Database Automation	Containers	Monitoring
Continuous Integration	Release Orchestration	Security
Testing	Cloud	Collaboration
Configuration	AIOps	

1 Os Gl GitLab																		2 En Sp Splunk
3 Fm Gh GitHub	4 En Dt Datical												5 En Xlr XebiaLabs XL Release	6 Fm Aws AWS	7 Pd Az Azure	8 En Gc Google Cloud	9 En Op OpenShift	10 Fm Sl Sumo Logic
11 Os Sv Subversion	12 En Db DBMaestro												13 Os Dk Docker	14 En Ur UrbanCode Release	15 Pd Af Azure Functions	16 Pd Ld Lambda	17 Fm Ic IBM Cloud	18 Os Fd Fluentd
19 En Cw ISPW	20 En Dp Delphix	21 Os Jn Jenkins	22 Fm Cs Codeship	23 Os Fn FitNesse	24 Fr Ju JUnit	25 Fr Ka Karma	26 Os Su SoapUI	27 En Ch Chef	28 Fr Tf Terraform	29 En Xld XebiaLabs XL Deploy	30 En Ud UrbanCode Deploy		31 Os Ku Kubernetes	32 Fm Cc CA CD Director	33 En Pr Plutora Release	34 Pd Al Alibaba Cloud	35 Os Os OpenStack	36 Os Ps Prometheus
37 Os At Artifactory	38 En Rg Redgate	39 Pd Ba Bamboo	40 Fm Vs VSTS	41 Fr Se Selenium	42 Fr Jm JMeter	43 Os Ja Jasmine	44 Pd Sl Sauce Labs	45 Os An Ansible	46 Os Ru Rudder	47 En Oc Octopus Deploy	48 Os Go GoCD		49 Os Ms Mesos	50 Pd Gke GKE	51 Fm Om OpenMake	52 Pd Cp AWS CodePipeline	53 Os Cy Cloud Foundry	54 En It ITRS
55 Os Nx Nexus	56 Os Fw Flyway	57 Os Tr Travis CI	58 Fm Tc TeamCity	59 Os Ga Gatling	60 Fr Tn TestNG	61 Fm Tt Tricentis Tosca	62 Pd Pe Perfecto	63 En Pu Puppet	64 Os Pa Packer	65 Fm Cd AWS CodeDeploy	66 En Ec ElectricCloud		67 Os Ra Rancher	68 Pd Aks AKS	69 Os Rk Rkt	70 Os Sp Spinnaker	71 Pd Ir Iron.io	72 Pd Mg Moogsoft
73 Fm Bb BitBucket	74 En Pf Perforce HelixCore	75 Fm Cr Circle CI	76 Pd Cb AWS CodeBuild	77 Fr Cu Cucumber	78 Os Mc Mocha	79 Os Lo Locust.io	80 En Mf Micro Focus UFT	81 Os Sl Salt	82 Os Ce CFEngine	83 En Eb ElasticBox	84 En Ca CA Automic		85 En De Docker Enterprise	86 Pd Ae AWS ECS	87 Fm Cf Codefresh	88 Os Hm Helm	89 Os Aw Apache OpenWhisk	90 Os Ls Logstash

XebiaLabs
Enterprise DevOps

Follow @xebialabs

Publication Guidelines
Download

91 En Xli XebiaLabs XL Impact	92 Os Ki Kibana	93 Fm Nr New Relic	94 En Dt Dynatrace	95 En Dd Datadog	96 Fm Ad AppDynamics	97 Os El ElasticSearch	98 Os Ni Nagios	99 Os Zb Zabbix	100 En Zn Zenoss	101 En Cx Checkmarx SAST	102 En Sg Signal Sciences	103 En Bd BlackDuck	104 Os Sr SonarQube	105 Os Hv HashiCorp Vault
106 En Sw ServiceNow	107 Pd Jr Jira	108 Fm Tl Trello	109 Fm Sl Slack	110 Fm St Stride	111 En Cn CollabNet VersionOne	112 En Ry Remedy	113 En Ac Agile Central	114 Pd Og OpsGenie	115 Pd Pd Pagerduty	116 Os Sn Snort	117 Os Tw Tripwire	118 En Ck CyberArk Conjur	119 En Vc Veracode	120 En Ff Fortify SCA

Developers are like Goldilocks

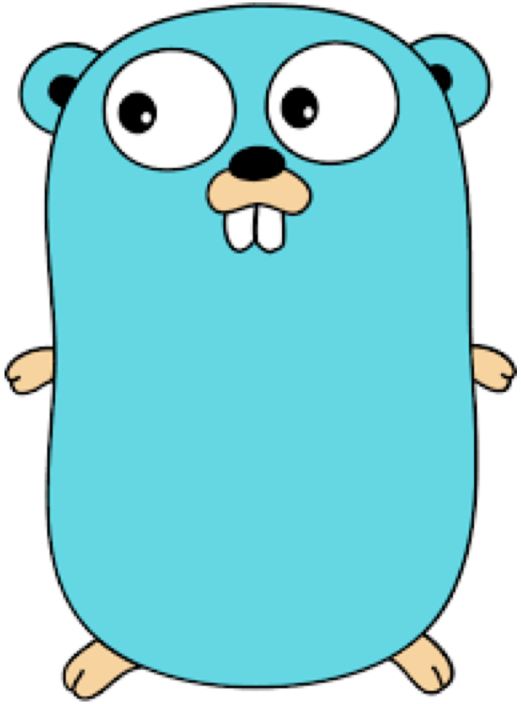
App Team

IaaS

PaaS

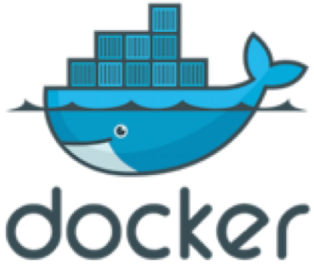
CaaS

Languages



What's on your Laptop?

- Development takes place on laptop
- Keep Local environment as close to production environment as possible
  - But still allow for desktop experience



Need a Place to Store Code

- A place to store current and past versions of code
  - Ability to merge, branch, fork
  - Ability to see who has changed which file
  - Highly available service
  - Ability to revert to prior versions
  - Public or Private / OnPrem / OffPrem
  - Social

GitHub

Bitbucket

GitLab

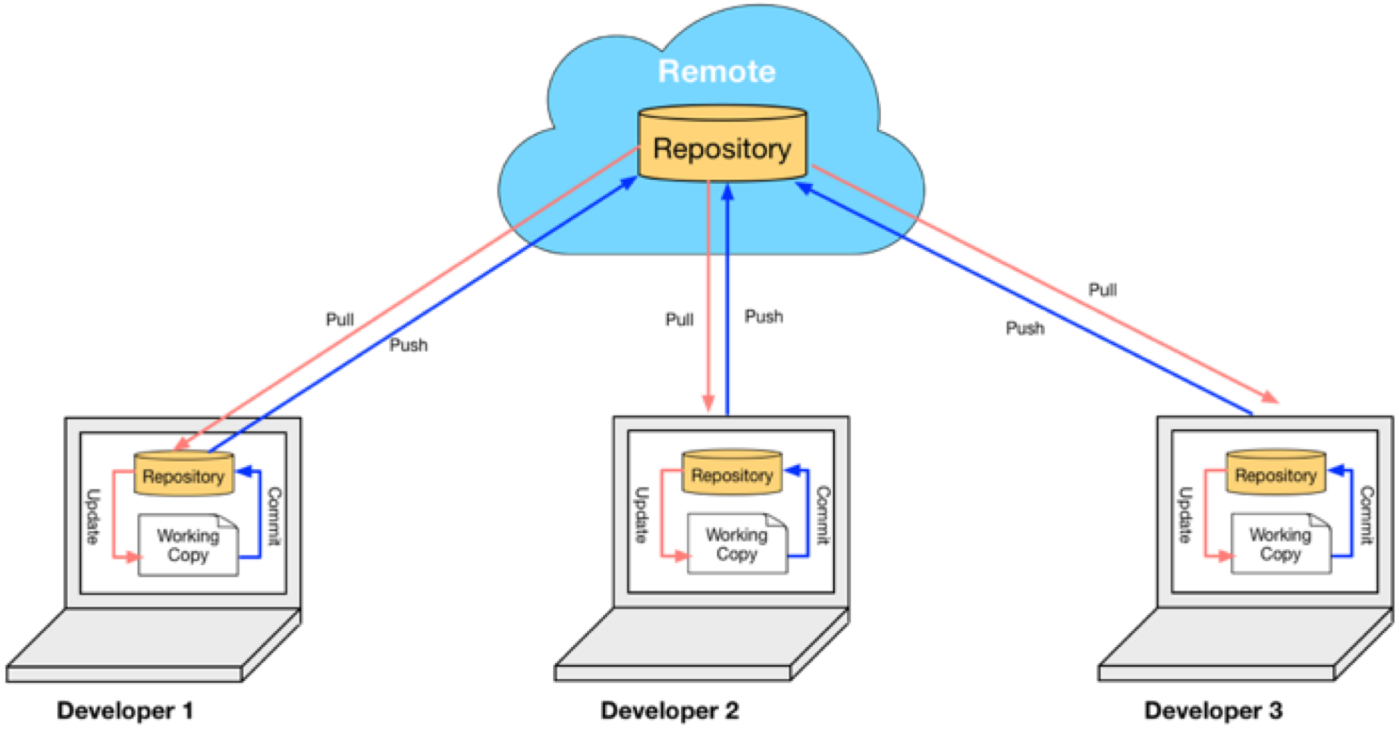
Don’t Check Passwords into Github

What is Git?

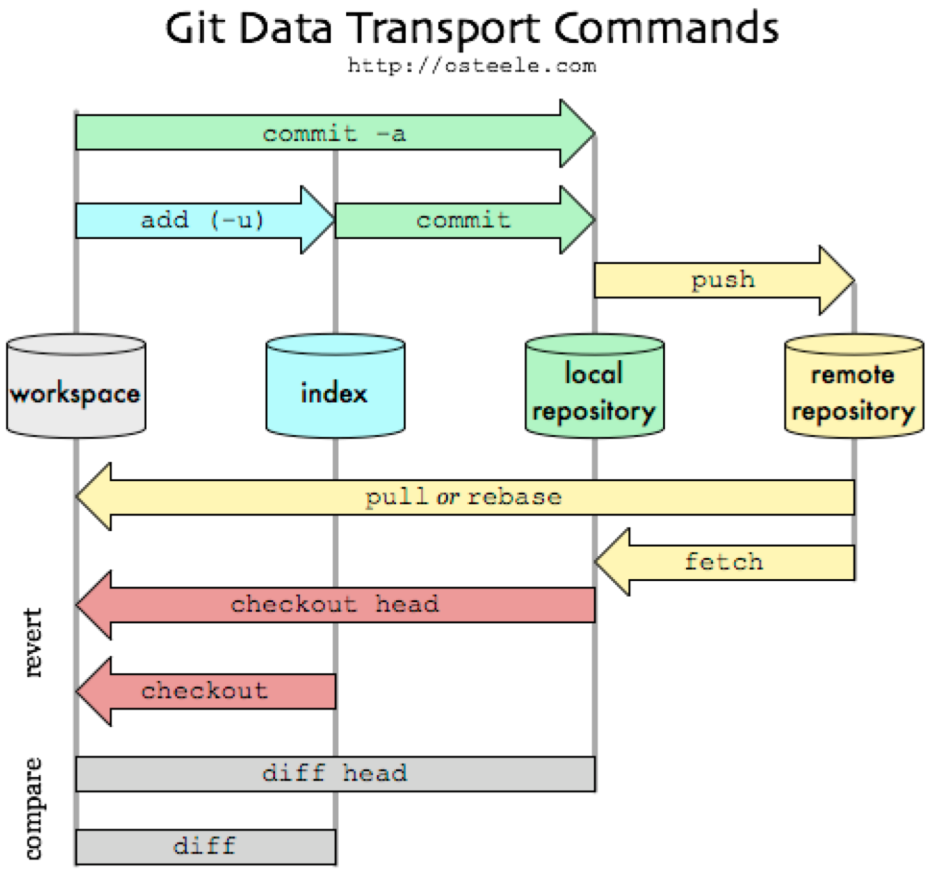
- **Git** (/gɪt/) is a **version control system** that is widely used for software development and other version control tasks. It is a distributed revision control system with an emphasis on speed, data integrity, and support for distributed, non-linear workflows.

- Git (software) - Wikipedia, the free encyclopedia

Git Distributed Version Control



Git Cheat Sheet



Pearson

Click to edit Master title style

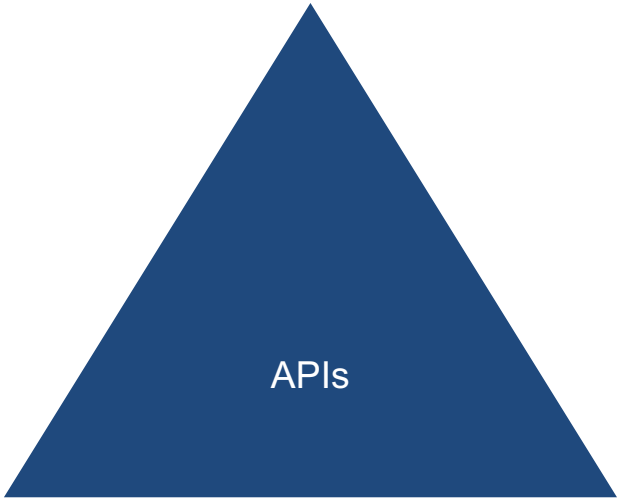
Break #4. 15 minutes

Editors

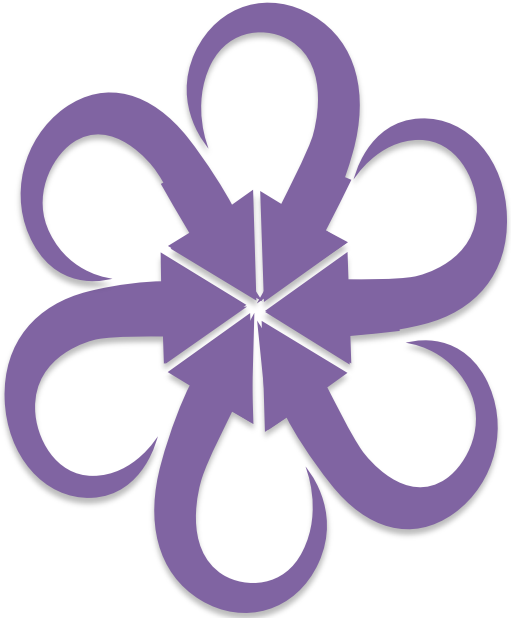
Collaboration

- Integration with Development Environment
  - Code commits should appear
- Chat with History
- Team Rooms
- Kanban Board
- Multi-Platform Clients
- “Chat-Ops”

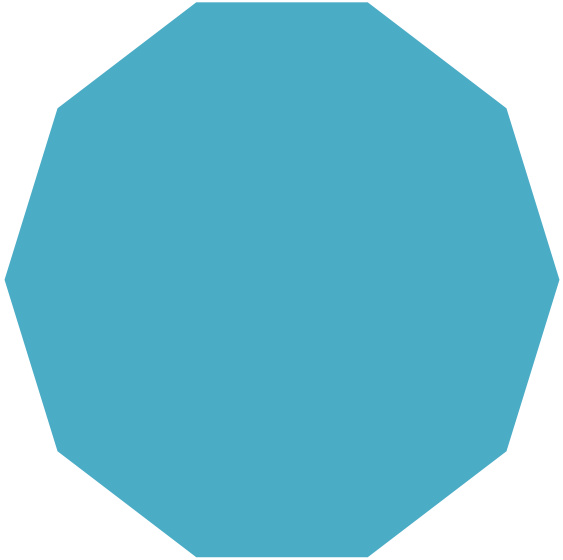
Need a Developer Platform



Programmable

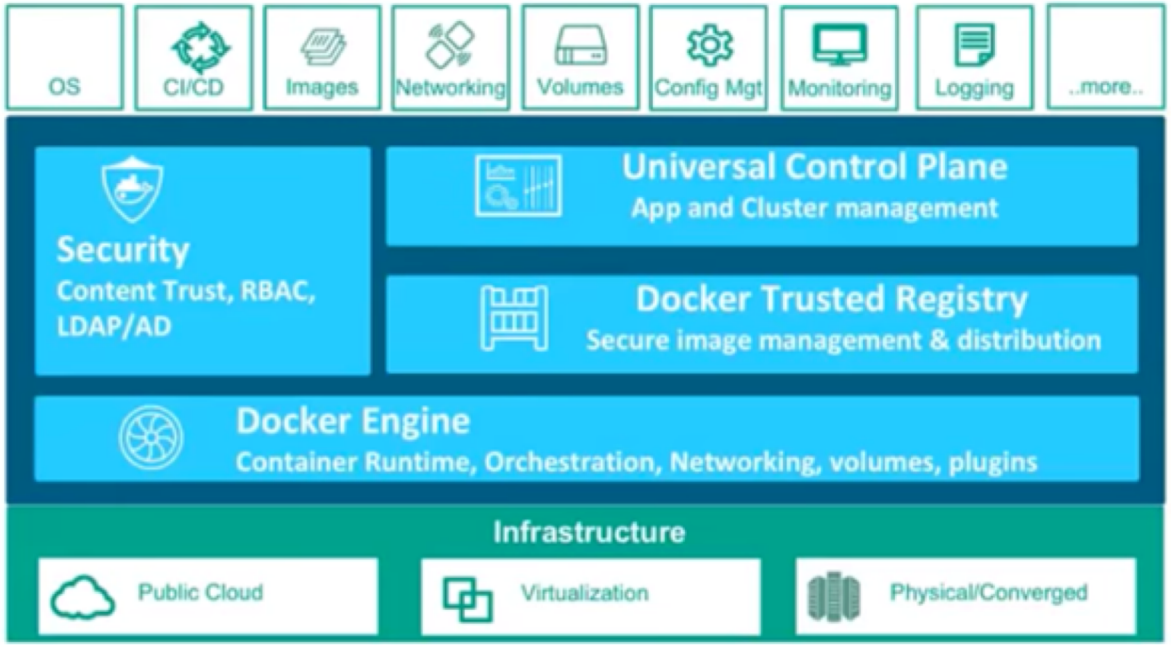


Tool Ecosystem



Available

Docker Datacenter



Platform as a Service

Multi-Cloud

Continuous Integration, Delivery, and

- Continuous Integration: Merging of development work with code base constantly so that automated testing can catch problems early.
- Continuous Delivery: Software package delivery mechanism for releasing code to staging for review and inspection.
- Continuous Deployment: Relies on CI and CD to automatically release code into production as soon as it is ready. Constant flow of new features into production

CI/CD Platforms: The Automated Software Bakery

Classic Mode

- Centralized Configuration
- Responsibility of CI/CD administrator
- Examples:
  - Jenkins, Microsoft TFS

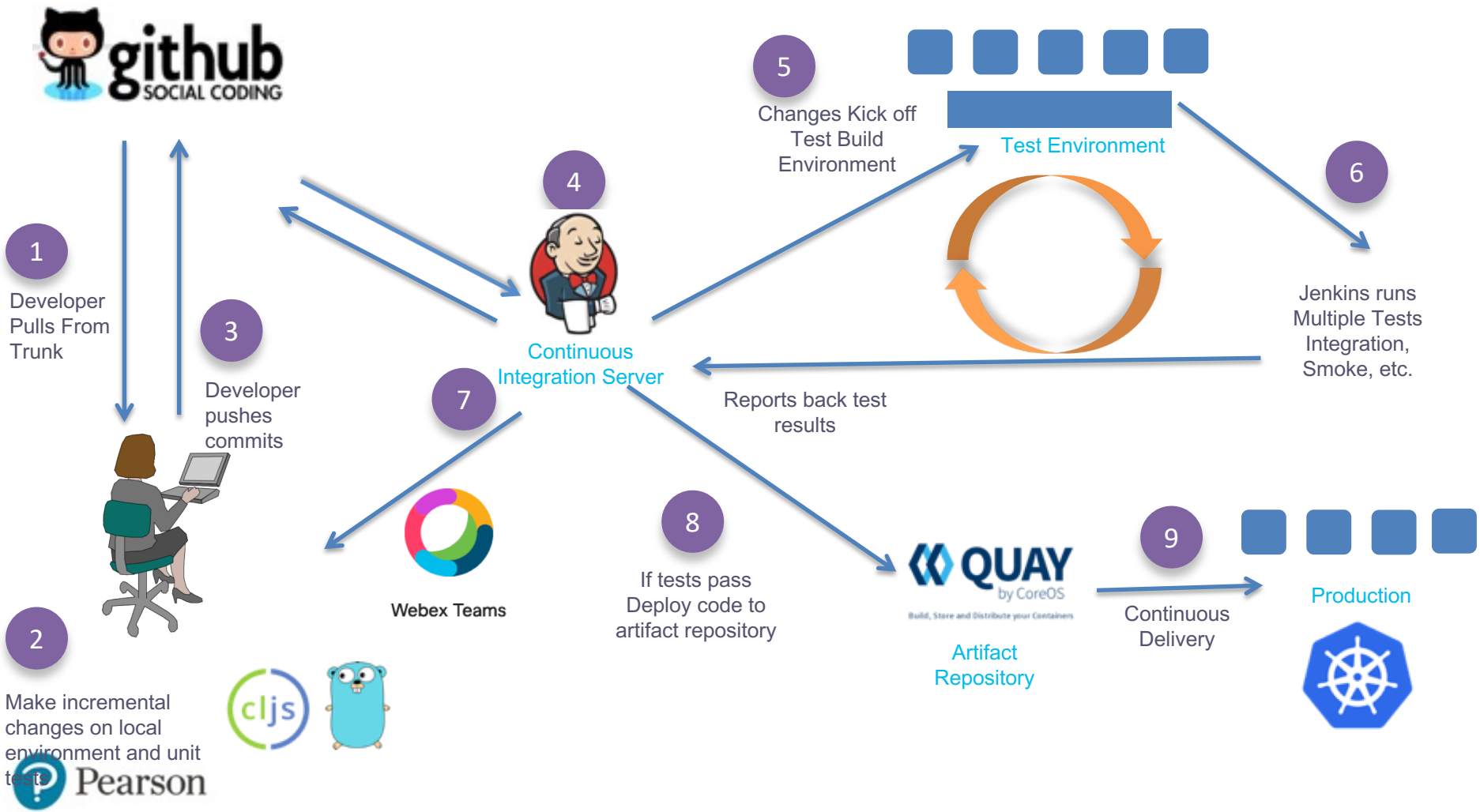
Emerging Mode

- Configuration pushed into the Repositories not centralized
- Responsibility of Developers
- Examples:
  - Drone.io, Travis CI, Circle CI

Monitoring, Logging, Alerting

Continuous Integration	
Configuration Management	puppet labs, Chef, SALTSTACK, ANSIBLE, heat
Collaboration	DIFFY
Working Environment	VAGRANT, PACKER, docker
Source/Image Control	GitHub, Bitbucket, docker, artifactory
PaaS	CLOUD FOUNDRY, OPENSHIFT
IaaS	openstack, amazon web services, Google Cloud Platform

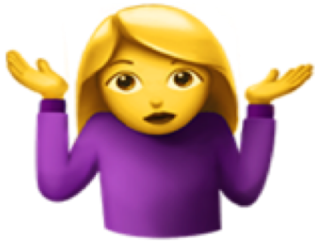
DevOps In Action


github
SOCIAL CODING
1
Developer Pulls From Trunk
2
Make incremental changes on local environment and unit test
3
Developer pushes commits
4
Continuous Integration Server
5
Changes Kick off Test Build Environment
Test Environment
6
Jenkins runs Multiple Tests Integration, Smoke, etc.
Reports back test results
7
Webex Teams
8
If tests pass Deploy code to artifact repository
QUAY
by CoreOS
Artifact Repository
9
Continuous Delivery
Production
cljs
Pearson

Demo: CI/CD with Kubernetes

What We Covered

What is this DevOps Thing?

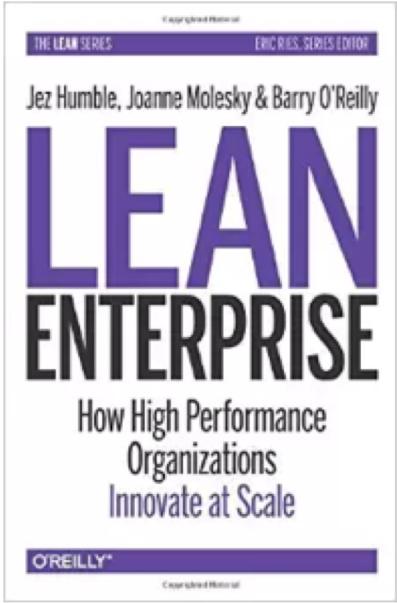


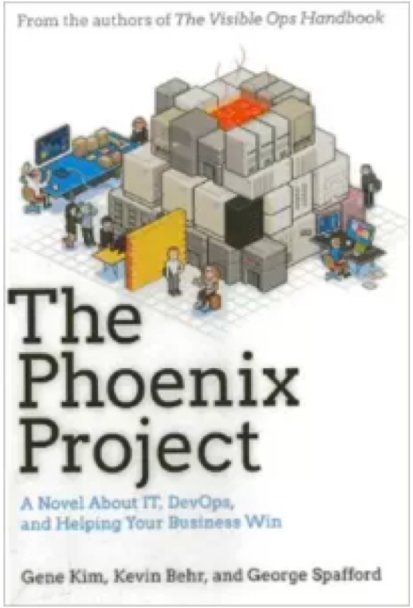
Why DevOps Matters

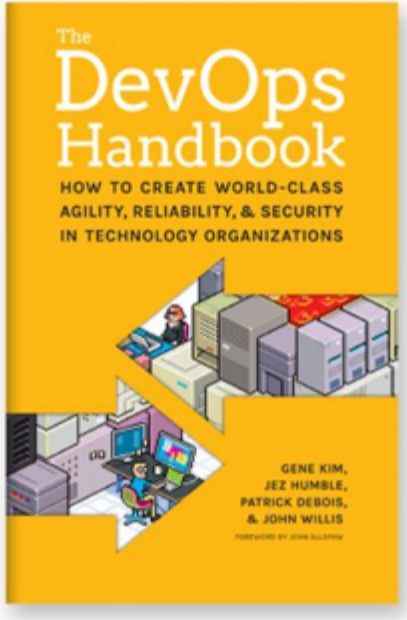
Going Faster with DevOps

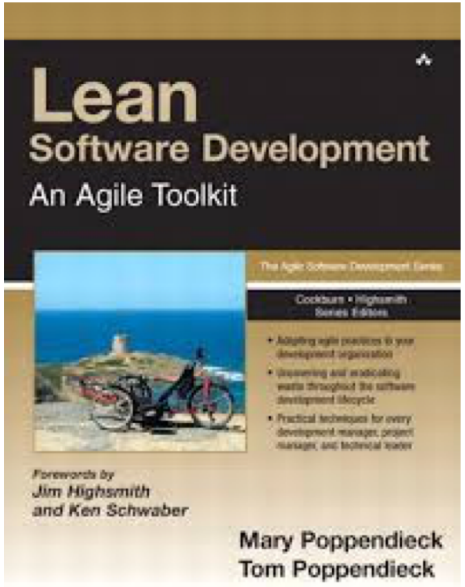
DevOps in Action

Further Reading









THANK YOU!